

Information Systems

Expert Systems

[INTERMEDIATE 2;
HIGHER]

James Bisset



The Scottish Qualifications Authority regularly reviews the arrangements for National Qualifications. Users of all NQ support materials, whether published by LT Scotland or others, are reminded that it is their responsibility to check that the support materials correspond to the requirements of the current arrangements.

Acknowledgement

Learning and Teaching Scotland gratefully acknowledge this contribution to the National Qualifications support programme for Information Systems.

First published 2004

© Learning and Teaching Scotland 2004

This publication may be reproduced in whole or in part for educational purposes by educational establishments in Scotland provided that no profit accrues at any stage.

ISBN 1 84399 029 6

CONTENTS

Section 1: Notes for staff	1
Section 2: Notes for students	7
Section 3: Teaching and learning materials	9
Chapter 1: What is an expert system?	9
Chapter 2: The knowledge base	25
Chapter 3: The inference engine	33
Chapter 4: The user interface	44
Chapter 5: Developing expert systems	54
Chapter 6: Creating an expert system	66
Chapter 7: Evaluating expert systems	75
Chapter 8: Further knowledge representation	80
Chapter 9: Further inferencing strategies	96
Chapter 10: Classical expert systems	104
Chapter 11: Implications of using expert systems	114
Glossary	125
Index	130
Section 4: Answers to exercises in Section 3	133
Section 5: Practical examples of expert systems	151



SECTION 1**Notes for staff****Aim**

This unit is designed to develop knowledge and skills in the use and construction of expert systems, and to meet the requirements of the unit specifications for Expert Systems at Intermediate 2 and Higher levels.

Status of this teaching and learning pack

These materials are for guidance only. The mandatory content of this unit is detailed in the unit specifications for the Expert Systems units in the relevant Arrangements documents.

Target audience

This is a bi-level teaching pack. It is intended for use with candidates working at Intermediate 2 and Higher levels.

Intermediate 2

While entry is at the discretion of the centre, students would normally be expected to have attained one of the following (or equivalent experience):

- Computing Studies course at Intermediate 1 level
- Standard Grade in Computing Studies (Grade 3 or 4).

Higher

While entry is at the discretion of the centre, students would normally be expected to have attained one of the following (or equivalent experience):

- Information Systems course at Intermediate 2 level
- Standard Grade in Computing Studies (Grade 1 or 2).

Pre-knowledge and skills

It is expected that students can make use of on-line help facilities as well as supporting software documentation and manuals.

Progression

This unit follows on from a study of database systems:

- database systems are used to store and organise information; expert systems are used to store information and to advise on a course of action.
- database systems can search for the results of a query; expert systems can search for advice to suggest.

The completion of this unit at Intermediate 2 level will enable students to embark on the Expert Systems unit at Higher level. The completion of this unit at Higher level will enable students to develop the concept of intelligent database systems, suitable for project work at Advanced Higher level.

Learning and teaching approaches

The materials in the pack are not intended to be used without teacher or lecturer input. For the study of the principles of expert systems it will be necessary for the teacher or lecturer to provide instruction and guidance at several points throughout the unit. For the practical study of expert systems an expert system shell will be required to deliver this unit, with supporting documentation in its use. However, it is helpful if candidates have exposure to at least one alternative expert system shell.

Appendix A at the end of Section 5 (pages 176–184) contains knowledge that must be converted into expert systems before the material earlier in Section 5 can be used. The files that are created will provide students with access to different types and varying complexities of expert systems. These expert systems will cover the content requirements for Outcome 2.

Pathway through the unit

Section 2 of this pack contains Notes for Students. Section 3 contains the main body of Teaching and Learning Materials that can be issued to students either together with or separately from the Notes for Students. Section 3 is subdivided into chapters, and contains a glossary of terms and index.

Section 5 contains practical exercises in the use of expert systems.

It is recommended that Section 3 is covered in parallel with Section 5, in which the practical examples can be used to illustrate the theoretical content. The implementation of the practical examples for any particular expert system shell will determine the extent to which these exemplify the theoretical content.

Hardware and software requirements

Centres are expected to have an expert system shell with an appropriate hardware configuration. Although there are examples of code included for illustration purposes, no particular shell has been exemplified in the pack and so the choice of tool is at the discretion of the centre.

Appendix A contains factor tables representing the knowledge required to complete the practical exercises in Section 5. These must be converted to expert system files in the chosen shell.

Centres are free to use this code for educational purposes, and may adapt the code as necessary provided the copyright notice within the file is retained.

General notes

These notes have been based on the original support notes provided by the Higher Still Development Unit for the Higher Expert Systems unit. Teachers familiar with those materials will recognise much of the text, illustrations and exercises, which now form a large part of the Intermediate 2 level materials.

The practical study of expert systems is dependent to a greater or lesser extent on the available expert system shell. At the time of writing, the most widely used expert system shells in schools are InterModeller® and flex®. Both are commercial products and are available for PC and Macintosh platforms.

There are a wide variety of free expert system shells available. However, centres should note that these are very often research vehicles and are provided 'as is', without support, and may be aimed at a university level audience. Nevertheless, some shells have active user groups that can provide useful backup (this is particularly true of CLIPS and JESS). The list on page 9 mentions some of the main expert system shells available.

Further information about available commercial and free expert system shells can be found at the Carnegie Mellon University (CMU) AI Repository. At the time of writing, the web address is:

<http://www-2.cs.cmu.edu/afs/cd.cmu.edu/project/ai-repository/ai/html/0.html>

References

During the writing of this material, reference was made to the following texts:

Bratko, I, *Prolog: Programming for Artificial Intelligence* (Addison Wesley, 1990)

Giarratano, J and Riley, G, *Expert Systems: Principles and Programming* (PWS Publishing Co, 1994)

Jackson, P, *Expert Systems* (Addison Wesley, 1990)

Rich E and Knight K, *Artificial Intelligence* (McGraw Hill, 1991)

These texts provide appropriate background material for teachers and lecturers but are not recommended as student texts.

Name of shell	Source/vendor	Web address	Platform(s)
InterModeller	Parallel Logic Programming	http://www.parlog.com	PC, Mac
flex	Logic Programming Associates	http://www.lpa.co.uk	PC, Mac
CLIPS (C Language Integrated Production System)	NASA (Johnson Space Center)	http://www.ghg.net/clips/CLIPS.html	PC, Mac, UNIX
JESS (Java Expert System Shell)	US Sandia National Laboratories	http://herzberg.ca.sandia.gov/jess	PC, Mac, UNIX
MIKE (Micro Interpreter for Knowledge Engineering)	Open University (no longer supported)	ftp://hcr1.open.ac.uk:/pub/software/ pc/MIKE25.ZIP	PC
D3	University of Würzburg	http://d3.informatik.uni-wuerzburg.de	Mac, PC (run time only)



SECTION 2**Notes for students****Introduction**

This unit is designed to develop your knowledge and skills in the use and construction of expert systems at Intermediate 2 and Higher levels. Before starting this unit, you should be able to make use of on-line help facilities as well as software documentation and manuals.

Information about this unit

This unit may be taken in combination with other units as part of the Information Systems courses at Intermediate 2 and Higher levels, or as a stand-alone unit. It is also possible for this unit to contribute to a Scottish Group Award.

The unit has two outcomes:

1. demonstrate knowledge and understanding of the principles, techniques and applications of expert systems
2. demonstrate practical skills by applying knowledge and understanding of the principles, techniques and applications of expert systems using contemporary hardware and software.

What will I learn during the unit?

By completing this unit, you will increase your knowledge of expert systems and their characteristics. You will also increase your practical skills in the use of an expert system shell.

The teaching and learning pack

This pack contains material to cover the contents of the unit. Section 3 covers the content of Outcome 1. This material will discuss the principles, techniques and applications of expert systems, including the processes involved in creating and evaluating an expert system. Section 5 covers the content of Outcome 2. This material will enable you to solve problems using expert systems. The expert systems that

you will use will be provided by your teacher or lecturer. You will be expected to learn about the structures available in the shell that you will use to create your own expert system.

What assessments will I have to take?

There will be two unit assessments:

- a 45-minute multiple-choice assessment to test your knowledge and understanding of the unit's contents
- evidence of your practical ability in the analysis, design, implementation, use and evaluation of an expert system.

SECTION 3

This section contains chapters relevant to the Intermediate 2 and Higher courses. Higher level material is contained within the shaded areas. Exercises are included for both levels, and are indicated as (Int2) or (H) as appropriate.

Chapter 1: What is an expert system?

Here are some dictionary definitions of an *expert system*:

- a computer system that asks questions and gives answers that have been thought of by a human expert (*Cambridge International Dictionary*)
- computer software that attempts to mimic the reasoning of a human specialist (*Webster's*)
- a computer system or program that uses artificial intelligence techniques to solve problems that ordinarily require a knowledgeable human (*encyclopedia.com*).

Each of these definitions tells us something interesting about an expert system:

- it is a computer program or piece of software
- it works by asking questions and giving answers
- it attempts to do something that is 'intelligent'
- it attempts to replace a human expert.

Before looking at expert systems in further detail, it is worth thinking about what we mean by an 'expert'. Again, here are some dictionary definitions of an *expert*:

- a person who is well informed or skilful in a subject (*Oxford English Dictionary*)
- a person with the special skill or knowledge representing mastery of a particular subject (*Webster's*)
- a person having a high level of knowledge or skill; a specialist (*Cambridge*).

We can see from these definitions that *experts* are all *humans*, while *expert systems* are *computer programs*. Also, experts are very highly

skilled people, which makes them special. That means there are not many experts around!

What are expert systems used for?

Expert systems are an application of the branch of computing science called 'artificial intelligence'. This field of study attempts to produce computer programs to perform tasks that, at the moment, humans do better. Expert systems are designed to solve problems and give advice that usually requires a human expert.

Here are some of the tasks for which expert systems have been produced:

- doctors use expert systems to help diagnose specialist diseases such as cancer or skin infections
- oil companies use expert systems to advise on the most likely places to drill for oil
- building societies use expert systems to help advise customers on the best type of savings account
- airlines use expert systems to help the pilot to diagnose problems with the aircraft and to plan the flight
- power stations use expert systems to advise what to do in the event of an emergency
- legal departments of large companies use expert systems to advise on certain aspects of the law
- British Gas uses an expert system to predict where corrosion is most likely to occur in the gas supply system
- Standard Telecommunications Company uses an expert system to design fibre optic cable
- GEC Marconi uses an expert system to identify faults in complex circuit boards.

Each of these different areas of use is known as the *domain of expertise* of the expert system. A human expert who has a great deal of knowledge about the area of use is known as a *domain expert*.

At Higher level, you will study some of the most important expert systems that have been produced (see Chapter 10).

Exercise 1.1 (Int2)

1. Identify the domain of expertise for which each of the following human experts is well known:
 - (a) Jamie Oliver
 - (b) Gary Lineker
 - (c) Jeremy Clarkson
 - (d) David Attenborough
 - (e) Jonathan Ross
 - (f) Tim Henman
 - (g) Patrick Moore
 - (h) Stephen Hawking
 - (i) Your teacher!

2. Identify a domain expert for the following domains of expertise:
 - (a) gardening
 - (b) interior design
 - (c) weather forecasting
 - (d) popular music.

3. Identify a domain of expertise of your own choice and a human expert for that domain.

4. Identify a domain of expertise for which you are a domain expert!

Defining an expert system

It is time now to introduce a more precise definition of an expert system to use in this unit.

Definition

An expert system is a computer program that:

- contains the specialist knowledge of one or more human experts; this expert knowledge is in a form that others may use to solve problems in a specific domain
- provides the user with advice via a consultation
- can explain the advice it gives and why it is asking particular questions.

So an expert system has three functions. Let us consider each of these in turn.

Function 1

An expert system contains the specialist knowledge of one or more human experts; this expert knowledge is in a form that others may use to solve problems in a specific domain.

For example, an expert system on cookery would contain specialist knowledge about that particular subject area or knowledge domain. It is likely that human experts such as Jamie Oliver and Delia Smith would be asked to provide the specialist knowledge contained in such an expert system. Users who are non-specialists in the area of cookery would be able to use such an expert system to plan meals and receive advice on any cookery problems that they may have.

Function 2

An expert system provides the user with advice via a consultation.

To use an expert system, a user carries out a *consultation*. When a user consults an expert system, the expert system will ask questions based on the specialist knowledge it contains. The expert system compares the user's answers with that knowledge and, once it has enough information, the expert system will then provide the user with output. The output from an expert system may be a diagnosis of what is wrong; it may be advice on what action to take or decision to make; it may provide a sequence of operations to perform to assist in planning; or it may help categorise new discoveries and findings. Expert systems are

often referred to as consultation systems because they provide advice much as human consultants do.

Function 3

An expert system can explain the advice it gives and why it is asking particular questions.

At any time during the consultation, the user can ask the expert system for an explanation of why a particular question is being asked. When the output is produced, the user can ask the expert system to explain how it arrived at that particular piece of advice.

Components of an expert system

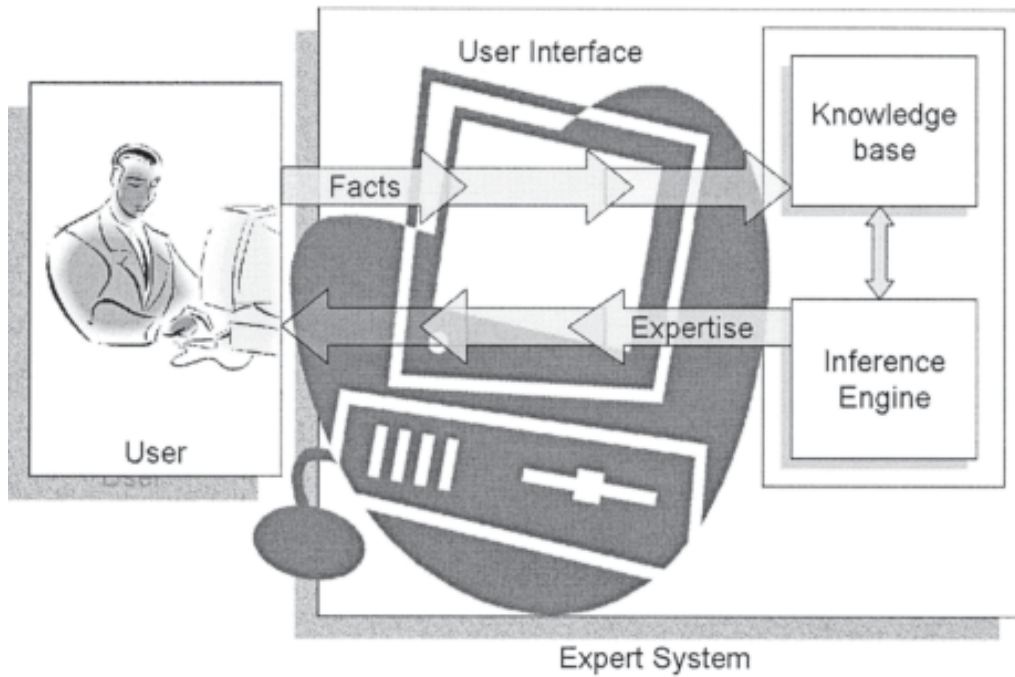
Expert systems work by using *facts* and *rules*. A fact is something like 'It is raining today' or 'the temperature = 21°C'. A rule draws a conclusion from facts, such as 'If it is raining today, take a brolly' or 'If the temperature = 21°C then switch the heating off'.

Expert systems are composed of three main components:

- the *knowledge base* which contains the facts and rules representing the domain knowledge (also called the 'rule base')
- the *inference engine* which applies the rules in order to reach a conclusion
- the *user interface* which displays questions and obtains answers from the user, displays conclusions and explains its reasoning.

Figure 1.1 shows how a user interacts with an expert system.

Figure 1.1: Components of an expert system



Each of the components of an expert system is described in further detail in the following chapters.

What is an expert system shell?

Expert system software is rarely created from scratch. More usually, it is created using some software called an *expert system shell*. An expert system shell is software that allows domain knowledge to be entered, and provides the reasoning capability and user interface

Most expert systems are developed using an expert system 'shell'. An expert system shell is an 'empty' expert system – it is empty because it contains no facts and rules in its knowledge base – but it does have a built-in user interface as well as a method of inferencing.

Definition

An *expert system shell* has an empty knowledge base, a user interface and an inference engine.

Data, information and knowledge

Before continuing, it is worth reviewing some definitions. *Data* is raw, unprocessed facts and figures. *Information* is the result of data that has been processed and has meaning, structure and context.

Expert systems deal with knowledge, so it is important to understand what makes knowledge different from information. *Knowledge* is created by a logical process in which new facts are derived from known facts by the application of *inference rules*. Put simply, having knowledge is the ability to use known facts to draw conclusions and make decisions.

Expert systems as information systems

Expert systems are sometimes referred to as *knowledge-based systems*. There is one significant difference between the two. A knowledge-based system may not simulate the reasoning of a human expert.

For example, a knowledge-based system is able to provide information on the weather based on the knowledge it contains about different types of weather. A meteorology expert system, on the other hand, may be able to predict the weather forecast by applying reasoning to the knowledge it contains.

Differences between expert systems and other information systems

What are the main differences between an expert system and other information systems such as decision-support systems (DSS), management information systems (MIS), and executive information systems (EIS)?

An expert system may be used at any level of an organisation where a human expert is normally required. For example, at an operational level, a car mechanic may require an expert to assist with diagnosing what is wrong with a particular vehicle; whereas, at the strategic level, an expert system may advise a managing director in making strategic decisions affecting the future of the company.

There is a similarity between expert systems and decision-support systems as both may draw on many different sources of information to form the knowledge base of the system, including databases and spreadsheets.

Databases and expert systems

Database systems provide the base for most of the information systems described above—DSS, MIS, EIS. They can also be linked with expert systems.

- Database systems store and retrieve large quantities of structured information.
- Human experts also tend to base the decisions they make on large amounts of information they have acquired.

- Database systems can be used to store the facts which are used within the expert system.
- Expert systems can then apply the rules within their knowledge base to the facts stored in the database system.

Deductive databases

A **deductive database** combines the storage power and capacity of a database system with the intelligence of an expert system. They are used in 'data mining' applications where an expert system attempts to deduce (or discover) useful information from the vast quantities of data being generated.

Expert systems and other programs

What is the main difference between an expert system and any other type of computer software?

It is the way it achieves its solution.

Most software follows **algorithms**. An algorithm is a sequence of steps that can be guaranteed to achieve a solution. Expert systems are different in that they attempt to find a solution by simulating human reasoning and use **heuristic** methods which are *not* guaranteed to succeed.

A heuristic is the process of gaining knowledge or some desired result by intelligent guesswork rather than by following some pre-established formula.

It is a 'rule of thumb' which guides the system towards a solution rather than guaranteeing it. A heuristic method has the advantage that it can still work on data that is imperfect or incomplete, and find solutions which are approximate.

Categories of expert systems

All expert systems produce advice as output but the way that the output is used determines the category of the expert system. There are several ways of categorising the output provided by an expert system. In this unit, we shall consider four different ways. These are described below.

The output provided by expert systems can be used for:

- advice
- classification
- diagnosis
- planning.

Expert systems used to give advice

If an expert system is used to give information to the user, then the output from such an expert system is used to *advise* on a course of action.

For example:

- companies with no legal department use expert systems to advise on certain aspects of the law
- oil companies use expert systems like GeoQuest and GeoPlay to advise on the most likely places to drill for oil
- British Gas use expert systems to predict where corrosion is most likely to occur in the gas supply system
- power stations use expert systems to advise what to do in the event of an emergency – in this case, the advice generated by the expert system is often used to confirm the advice given by a human expert
- some building societies use expert systems to assist in providing advice to customers on which type of savings account to open.
- American Express use an expert system called Authorizer's Assistant to help detect the attempted fraudulent use of credit cards.

Expert systems earning credit

Senior executives at American Express were convinced that artificial intelligence (AI) had the potential to help the company to improve some of its operations. Specifically, they felt the authorisation of credit-card purchases, a very time- and people-intensive process, could be handled by some form of AI.

The criteria for an 'authoriser's assistant' were fairly straightforward. It had to minimise fraud and credit losses from improper or incorrect authorisations. It had to assist authorisers in making more accurate authorisations more frequently and more quickly.

The first prototype took almost six months to complete and consisted of 520 rules (over the years, it has grown to around 2500 rules). The Authorizer's Assistant was transparent to Amex's users, who still used the same terminals and system software that were in place before the expert system project began. For them, it appeared as if the regular system had just taken on a new level of intelligence.

The regular system, however, was linked into a network of computers that were responsible for running an expert system. There, authorisation decisions were made and sent back to the mainframe, which then returned the approval or disapproval over the phone lines to the original merchant and card holder.

American Express claims it has saved tens of millions of dollars per year using the Authorizer's Assistant. It does the work of 700 authorisation employees, many of whom have been transferred to other, less routine job functions.

Expert systems and helpdesks

Every day thousands of people make use of helpdesks. Nowadays, these helpdesks are intelligent. In other words they make use of expert systems that allow the customer service representative to input the customer's questions or problems into a computer system. The system will then search its knowledge base to find the correct solution. This system is far more efficient than a human manual system, which could not possibly remember all the types of problems that occur every day.

Both customers and employers benefit from these systems as these smart assistants reduce phone time and increase customer satisfaction.

Expert systems used for classification

If an expert system is used to identify or grade information, then the output from such an expert system is used to *classify* information.

For example:

- botanists use expert systems to classify rare plants
- geologists use expert systems to categorise minerals
- *Thesys* is an expert system that is used in some universities to allow students to grade their own written reports before submitting them
- expert systems are used by many supermarket chains to identify patterns of customer spending.

Expert systems and the human expert

Coopers and Lybrand needed to develop a computerised method of assisting with company audits and changes in the tax structure.

In one year 24 experts contributed a great deal of their time to create the ExperTax expert system at a cost of \$1 million. The system was set up in all 96 of Coopers and Lybrand's US offices. In other words, Coopers and Lybrand had 96 computer clones of its experts in each office to deal with tax strategies.

Expert systems keep it on the road

The Ford Motor Company uses the Service Bay Diagnostic System (SBDS), which addresses the problems of dealer servicing of cars still under warranty.



Due to the increasing complexity of newer model car components, many car technicians or mechanics were finding it easier to swap out entire car sub-assemblies rather than find a specific component that needed to be repaired. Thus, a sub-assembly consisting of numerous components, instead of an individual component, was thrown away, even when much of the sub-assembly was in working order.

This increased the cost of warranty repair to Ford because the cost of all the parts was passed to the car manufacturer by the dealer.

Ford wanted to have more detailed repairs carried out with fewer parts returned, thereby keeping warranty expenditures down. The idea behind the system was to put a PC with a 20000-rule diagnostic expert system in every dealership service bay and to have it guide mechanics through the diagnosis and repair procedure. This way, a more precise repair could be made, eliminating the wholesale swapping out of parts.

It also minimised the time it took for the repairman to find the problem, which meant the customer got the car back sooner, which in turn meant Ford looked better when it came time to publish the latest customer satisfaction ratings. And since the engines on new cars were beginning to look like the inner workings of an atomic reactor, the system assisted mechanics in staying current with changes in automotive design.

An interesting side-effect of this project was that the expert system developers discovered an unusual fact about car mechanics: a significant proportion of them were dyslexic. This prevented those mechanics from getting more skilled engineering jobs. To address this, they created an intricate natural language interface to the expert system to make the system more accessible to Ford's mechanics.

Expert systems used for diagnosis

If an expert system is used to predict or determine the cause of a problem, then the output from such an expert system is used to help to *diagnose* problems.

For example:

- medical expert systems are used to diagnose illnesses and diseases
- government agencies are using expert systems for troubleshooting and intelligent alternative selection
- airlines use expert systems to help the pilot to diagnose problems with the aircraft
- GEC Marconi used an expert system called APEX to identify faults in complex circuit boards – in fact, diagnostic expert systems are widely used to help speed up the repair of complex equipment
- BP Chemicals Grangemouth uses an expert system for fault analysis in their butadiene plant

Expert systems used for planning

If an expert system is used to design or prepare an itinerary or schedule, then the output from such an expert system is used to help *plan*.

For example:

- the SUMit expert system was used by KLM airlines to plan employee rotas
- the expert system used by the Standard Communications Company is used to plan the design of fibre optic cables
- the expert system used by Lufthansa aircraft is used to plan the flight path for an aircraft's journey
- military expert systems help provide strategic advantage over the enemy by assisting in battle assessment
- manufacturing expert systems help companies to plan better work flow and identify areas of improvement
- the TRANS expert system is used to program space observations for the Hubble Space Telescope
- Nordic Offshore Systems uses a real-time expert system to monitor the drilling process on offshore oil-drilling platforms to warn of imminent danger from explosions due to heat, pressure, and vibration
- PowerGen uses the SHIRAS deductive database to plan shift patterns.

Exercise 1.2 (Int2)

1. Describe three functions of an expert system.
2. Where does the knowledge in an expert system come from?
3. An expert system entitled 'Parliament: How it Works' is being created. Identify two appropriate domain experts for this expert system.
4. Identify the three components of an expert system.
5. What two forms of output are generated from an expert system?
6. Write down the category of each of the following expert systems.
 - (a) An expert system that suggests what toy to buy for a child at Christmas.
 - (b) An expert system that identifies different types of yachts.
 - (c) An expert system that prepares an itinerary for a cycling holiday.
 - (d) An expert system that suggests what is wrong with a faulty monitor.
 - (e) An expert system that gives suggestions of jobs that the user might want to consider.
 - (f) An expert system that contains knowledge of radio stations and suggests which radio station a user might prefer to listen to.
 - (g) An expert system used in garden centres to suggest appropriate treatment for different types of soil.
 - (h) An expert system that identifies different types of transport.
 - (i) An expert system used by home economists to prepare special menus.
7. Consider the output produced by the expert systems below and determine the category of each expert system.
 - (a) 'You should buy Trade and Mart'.
 - (b) 'The rock is limestone'.
 - (c) 'You should write out a guest list before sending out the invitations'.
 - (d) 'The baby is crying because he is wet'.
 - (e) 'The spark plugs are faulty'.
 - (f) 'The creature is a mammal'.

8. Read the case study 'Expert systems earning credit' on page 23.
 - (a) Why did the managers of American Express want an expert system to help their company?
 - (b) Why do you think the prototype system took six months to complete?
 - (c) Why was it important that the system was 'transparent' to its users?
 - (d) Give two reasons why American Express were able to save 'tens of millions of dollars each year'.

9. Read the box 'Expert systems and helpdesks' on page 24.
 - (a) Why are expert systems needed to assist the human operators on a helpdesk?
 - (b) Give two benefits of the use of an intelligent helpdesk:
 - (i) to the customer
 - (ii) to the company.

Exercise 1.3 (H)

1. Read the box 'Expert systems and the human expert' on page 24.
 - (a) Why was an expert system needed by Coopers & Lybrand?
 - (b) How would the system designers know whether or not their system was successful?
 - (c) Why were so many experts involved in developing the system?
 - (d) What was the most expensive aspect of the development of the system?
 - (e) What were the benefits of the system to the company's US offices?
 - (f) Why did the company consider the cost of developing the system to be worth it?

2. Read the box 'Expert systems keep it on the road' on page 25.
 - (a) Why was an expert system needed by Ford Motor Company?
 - (b) Give one benefit to customers of the use of the expert system.
 - (c) Give two benefits to the company of the use of the expert system.
 - (d) Describe how the user interface of the expert system was adapted to suit its users.

Chapter 2: The knowledge base

The knowledge base contains the *facts* and *rules* that represent the specialist knowledge contained in the expert system. Here are some examples of facts and rules:

Facts

The bathroom is dry
The hall is wet
There is a leak in the kitchen

Rules

IF the bathroom is dry
AND the hall is wet
THEN there is a leak in the kitchen.

The most widely used form of knowledge representation in expert systems is the *production rule*. A production rule is usually expressed using the keywords IF and THEN, and has two parts: *conditions* and *actions*. The IF part states a condition (or premise) and the THEN part expresses a corresponding action or conclusion. Here are some examples:

IF animal produces live offspring (*condition*)
THEN the animal is a mammal. (*conclusion*)

IF the bathroom is dry (*condition 1*)
AND the hall is wet (*condition 2*)
THEN there is a leak in the kitchen (*conclusion*)

Sometimes production rules are expressed in the form <conclusion> IF <conditions>. For example:

The animal is a mammal (*conclusion*)
IF animal produces live offspring. (*condition*)

There is a leak in the kitchen (*conclusion*)
IF the bathroom is dry (*condition 1*)
AND the hall is wet. (*condition 2*)

The way the rules above are written is very similar to everyday English and is known as 'pseudo-code'. Pseudo-code rules cannot normally be entered directly into an expert system shell.

Multiple conditions in production rules

Production rules can have one or more conditions. Conditions can be combined using the three keywords AND, OR and NOT (these are known as Boolean operators¹).

Here are three examples, using each of the Boolean operators:

Rule 1 IF the bathroom is dry
 AND the hall is wet
 THEN there is a leak in the kitchen.

Rule 2 IF the kitchen is wet
 OR the hall is wet
 THEN there is a leak in the kitchen.

Rule 3 IF the bathroom is wet
 AND the hall is NOT wet
 THEN there is a leak in the bathroom.

It is also possible to combine conditions in more complicated ways using the Boolean operators. Here is an example:

Rule 4 IF the hall is wet
 AND (the bathroom is dry OR the kitchen is wet)
 THEN there is a leak in the kitchen.

This single rule combines rules 1 and 2 listed above.

Representing alternatives

Production rules that contain alternative conditions (linked with the OR operator) can be represented in most expert systems as a number of alternative rules each with the same conclusion. For example, the rule:

Rule 2 IF the kitchen is wet
 OR the hall is wet
 THEN there is a leak in the kitchen.

¹ Named after George Boole, 19th-century British mathematician and logician.

can be represented as the following two rules:

```

Rule 2a  IF      the kitchen is wet
          THEN   there is a leak in the kitchen.
Rule 2b  IF      the hall is wet
          THEN   there is a leak in the kitchen.

```

This method can also be used to avoid a rule becoming too large and complex.

Some expert system shells do not support the use of the OR operator. In this case, rules with alternative conditions must be represented as separate rules, as in the example above.

The production rules above are written in pseudo-code. To be entered into an expert system shell, they must be expressed in an appropriate *knowledge representation language* (KRL). Each expert system shell has its own KRL. Here are some examples of the production rules above expressed in a KRL²:

```

relation advise(mammal)
    if the reproduction is 'live offspring'.

```

```

Rule 1
    if the bathroom is dry
    and the hall is wet
    then the leak`s location becomes kitchen.

```

² The examples given are expressed in KSL (knowledge specification language), the KRL used by the LPA flex expert system shell.

Knowledge representation

Consider the following passage of text about different methods of transport:

There are three main modes of transport: land, sea and air.

Land-based locomotion is either by road or rail. Road-based locomotion includes cars, which are designed for carrying people, and trucks, which are designed for carrying freight. Trams and trains travel on rails, with trams designed for carrying people, while trains are designed for carrying people and freight.

Sea-based locomotion is by ship. Tankers are designed for carrying freight, while ferries are designed for carrying people.

Air travel is by aeroplane or helicopter. Aeroplanes are designed for carrying people and freight, while helicopters are designed for carrying people.

An expert system is required to classify methods of transport. How should the knowledge contained within the text be extracted and converted into production rules?

Factor tables

A useful technique to help structure the knowledge is to use a *factor table*. A factor table is simply a way of matching up conditions to actions or conclusions. Here is a factor table to represent the text above.

Transport	Mode	Cargo	Locomotion
Car	<i>Land</i>	<i>People</i>	<i>Road</i>
Truck	<i>Land</i>	<i>Freight</i>	<i>Road</i>
Tram	<i>Land</i>	<i>People</i>	<i>Rail</i>
Train	<i>Land</i>	<i>People & freight</i>	<i>Rail</i>
Tanker	<i>Sea</i>	<i>Freight</i>	
Ferry	<i>Sea</i>	<i>People</i>	
Aeroplane	<i>Air</i>	<i>People & freight</i>	
Helicopter	<i>Air</i>	<i>People</i>	

The factor table is similar to a table in a database. The column headings correspond to fields in a database table. The rows of the table correspond to records in a database table.

As in a database, the column headings are called *attributes*, with the *values* listed in the rows below. The factor table produces a set of *attribute–value* pairs, which classify each type of transport. For example, here is the set of attribute–value pairs for a car:

mode = land, cargo = people, locomotion = road

As most expert systems work by using attribute–value pairs to represent facts in the knowledge base, a factor table is a useful way of listing these.

From factor table to production rules

Once you have a completed factor table, it is a relatively straightforward process to convert the table into a set of production rules.

The *rows* of the factor table correspond to *rules*, and the *columns* correspond to *conditions* or *questions*. So, for the factor table above, there should be eight rules, with up to three conditions each. Here are the rules for car and tanker:

The method of transport is a car
 IF the mode is land
 AND the cargo is people
 AND the locomotion is road.

The method of transport is a tanker
 IF the mode is sea
 AND the cargo is freight.

Notice that the locomotion condition doesn't apply to some of the methods of transport, so the value has been left blank in the table. This means this condition can be ignored in the rules for tanker, ferry, aeroplane and helicopter.

(In some expert system shells³, the conditions also require a question to be entered, so that three questions would also require to be entered.)

There are other knowledge representations that can be used to structure and code knowledge. At Higher level, you are required to know how to use *decision trees*, *forward* and *backward chaining rules* and *logic*. These are covered in Chapter 8.

³ e.g. LPA flex

Structuring the knowledge

The purpose of a factor table is to ensure that the knowledge is represented correctly and efficiently. This means using the right number of rules and the correct conditions to ensure every conclusion can be reached. It is desirable to avoid unnecessary questioning, such as:

Q: Is the mode of transport by land?

A: No.

Q: Is the mode of transport by sea?

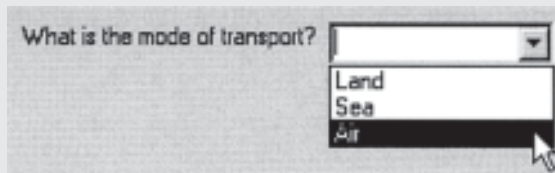
A: No.

Q: Is the mode of transport by air?

A: Yes.

This often arises if questions are posed with 'Yes/No' answers. It is generally preferable to avoid this type of 'closed' question in favour of 'open' questions with multiple responses. For example, a more efficient means of obtaining the mode of transport is to ask a single question and provide the user with the appropriate choices, as shown in Figure 2.1.

Figure 2.1: Using multiple answers to avoid closed questions

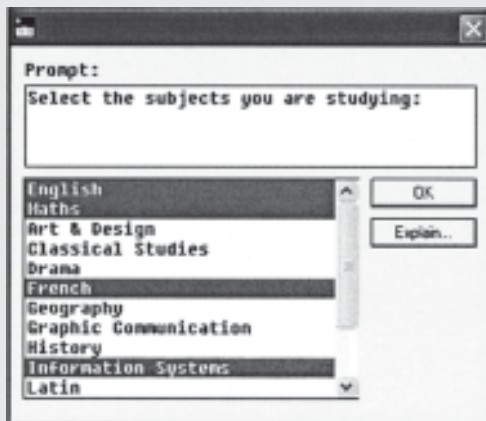


In the factor table, this means that instead of having three column headings for 'Land', 'Sea' and 'Air', with 'Yes' or 'No' entered on each row, we have one heading called 'Mode', with entries 'Land', 'Sea' and 'Air'.

Replacing closed questions in this way means that it is also possible to group certain conditions together if suitable.

In some expert systems, it is possible to ask questions that allow multiple answers to be selected, as shown in Figure 2.2.

Figure 2.2: Selecting multiple answers in the flex expert system shell



Note that it is possible for the cells in the factor table to contain blanks if a condition doesn't apply. In this case we would expect the expert system to ignore the attribute. (Of course, it could be that by providing a value for that attribute, the conclusion is no longer valid! If this is the case, then the expert system's rules must be altered to take account of this.)

As a general rule, if a factor table has Yes/No entries, or many entries that only apply to individual conclusions, it is worth considering if the table can be better structured to combine these under a single heading.

Exercise 2.1 (Int2)

1. Consider the following information:

Healthy food is full of vitamins as well as being full of protein.

Write a production rule in psuedo-code that matches this information.

2. *If the night sky is clear in wintertime, it usually means that there will be frost lying in the morning.*

Write a production rule in pseudo-code to represent the information above.

3. Represent the following information as a set of attribute–value pairs in a factor table.

Cirrus clouds are examples of high-level clouds. They are wispy clouds that indicate that it will rain soon.

Altostratus is a medium-level cloud that forms a thick layer, blocking out the sun although rain is unlikely.

Nimbostratus is a low-level cloud that forms a flat, featureless layer and indicates that rain is likely.

Chapter 3: The inference engine

Once knowledge has been represented in some form, the expert system needs to be able to use the knowledge to draw conclusions and give advice. What the expert system needs is a method of reasoning to do this.

The *inference engine* determines the method of reasoning used by the expert system. This means that it determines how the expert system applies its rules to the facts contained in the knowledge base. The order in which the rules are applied in turn determines the order in which questions are posed to the user.

There are two main methods of inferencing used by expert systems with production rules. These are *forward chaining* and *backward chaining*.

Depending on the design of the expert system, the inference engine may do either forward or backward chaining, or possibly even a combination of both of these. The choice of inferencing method depends on the type of problem being solved by the expert system.

Typically, advice, classification and diagnosis systems use forward chaining, and planning systems are better suited to backward chaining.

Backward chaining

Recall the 'household plumbing' rules introduced in Chapter 2. Here is a rule:

There is a leak in the kitchen
IF the bathroom is dry
AND the hall is wet.

In backward-chaining expert systems, we start by making a 'guess' as to the answer. This is called a *hypothesis* – for example, 'there is a leak in the kitchen'. We then work back from there seeking evidence to support our hypothesis.

In this example, to confirm the hypothesis, we need to find evidence that the bathroom is dry and that the hall is wet. In backward chaining, the expert system focuses on only one hypothesis at a time – all others are considered at that stage to be irrelevant – and sets about trying to find the evidence for that hypothesis.

If evidence cannot be found to support the hypothesis (e.g. the bathroom is not dry or the hall is not wet), then our original hypothesis was wrong and we must start again with an alternative hypothesis. This process is repeated until either a hypothesis is proven or there are no further hypotheses left to try.

Backward chaining is also known as *goal-driven chaining* because at each stage the system is trying to find evidence to support a hypothesis or achieve a goal.

Goal-driven backward chaining

To understand how backward chaining works, let's think about how to make breakfast. This could be represented using the following rule, with the goal 'make breakfast':

Rule 1 Make breakfast
 IF make cereal
 AND make toast
 AND make tea.

Each task (or 'sub-goal') in Rule 1 can be broken down into further sub-goals. This is called 'chaining'. For example, here is a rule to 'make cereal':

Rule 2 Make cereal
 IF cereal is cornflakes
 AND have milk.

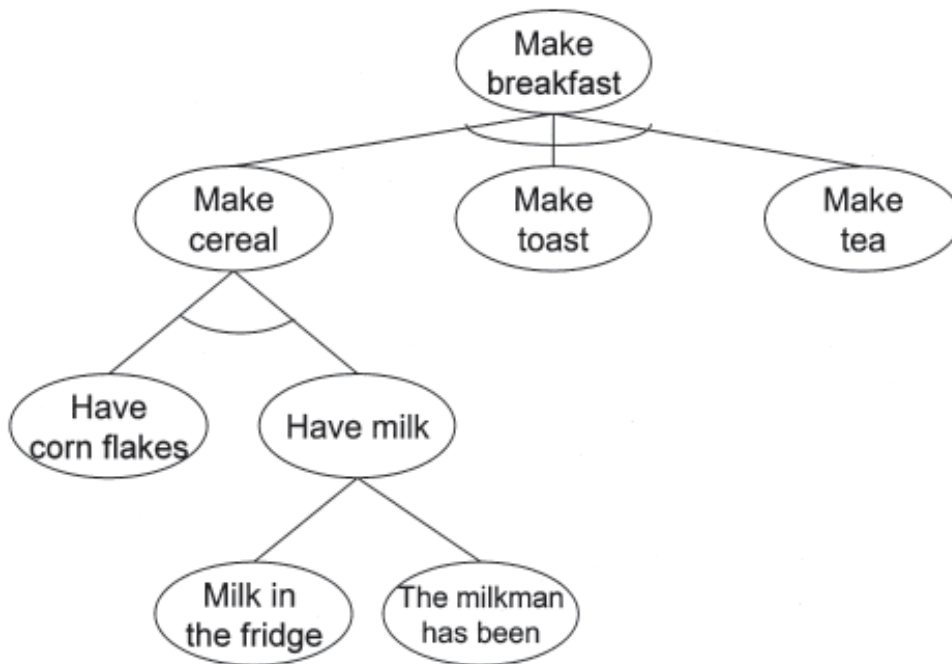
We could continue to break each of these tasks down further, as required. For example, here is a rule to achieve the sub-goal 'have milk':

Rule 3 Have milk
 IF milk in the fridge
 OR the milkman has been.

This type of rule is used mainly in planning systems that are goal-driven.

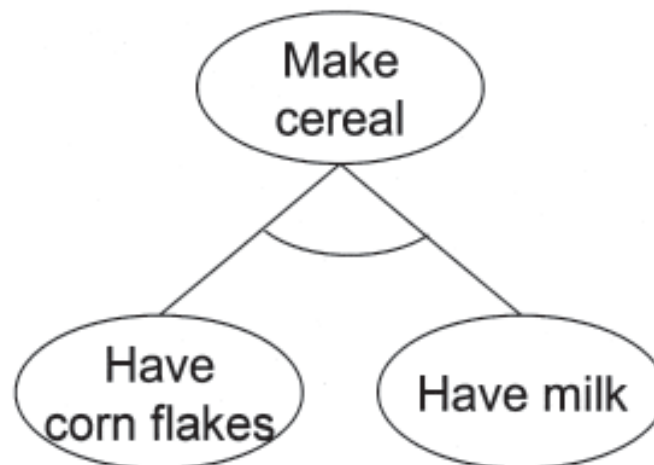
These levels of chaining can be represented in a diagram (called an AND-OR graph):

Figure 3.1: AND-OR graph



The lines are called 'branches' and indicate which sub-goals must be completed (or 'satisfied') in order to complete the goal above. So, completing the goal 'make cereal' needs the sub-goals 'have cornflakes' and then 'have milk' to be satisfied, as shown in Figure 3.2.

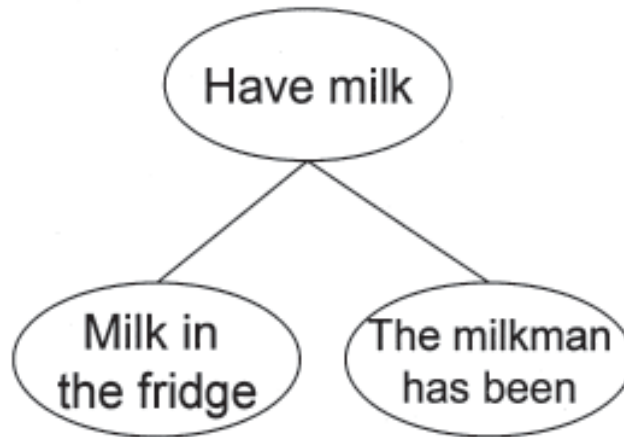
Figure 3.2: AND branches – the goal 'make cereal' with two sub-goals



The curved line joining these sub-goals together indicates that all the sub-goals must be completed.

The goal 'have milk' also has two sub-goals, but this time these are alternatives, as shown in Figure 3.3.

Figure 3.3: OR branches – the goal 'have milk' with two sub-goals



This means that to 'have milk' requires either that 'there is milk in the fridge' or that 'the milkman has been'. As long as one of these sub-goals is true, then the goal 'have milk' is also true.

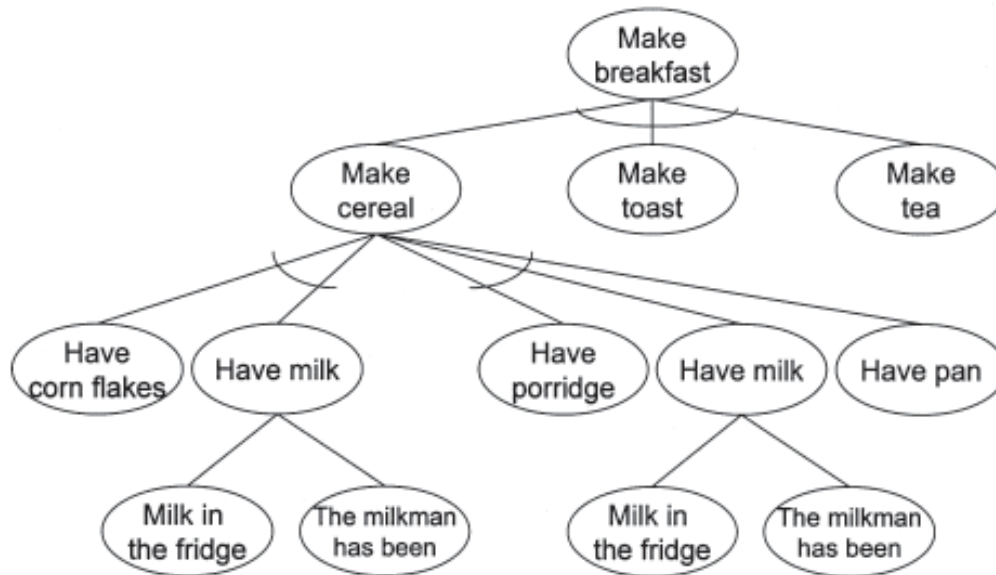
If any of the sub-goals cannot be satisfied, then the goal above cannot be satisfied. For example, if the sub-goal 'have cornflakes' cannot be satisfied (perhaps because there is only porridge in the cupboard!), then the goal 'make cereal' cannot be satisfied, and so the top-level goal 'make breakfast' cannot be satisfied.

Of course, the solution to this problem is to have an alternative rule for making cereal that applies to porridge. Here it is:

Rule 2a Make cereal
 IF have porridge
 AND have milk
 AND have pan.

The updated diagram is shown in Figure 3.4.

Figure 3.4: Extended AND–OR graph



Notice that there are now two alternative ways of satisfying the goal 'make cereal'. If the first attempt – making cornflakes – fails, then the next alternative – making porridge – will be attempted. This process is called *backtracking*. The system will continue backtracking until either a sub-goal is satisfied or there are no more alternatives to try.

Note that the order of the rules is very important in a backward-chaining system because this determines the order in which the hypotheses are attempted.

So if you don't like porridge you need to make sure your 'make cereal' rule for cornflakes comes before your 'make cereal' rule for porridge!

Exercise 3.1 (Int2)

1.
 - (a) Draw an AND–OR graph to show how to complete the goal ‘make toast’ shown in Figure 3.4. This involves toasting the bread, spreading margarine and spreading jam.
 - (b) Extend your graph for (a) to show two alternative ways of toasting the bread (using a toaster or using a grill) and alternative spreads for the top (butter or margarine, and jam, honey or marmalade).
 - (c) Extend your graph for (b) further to check to see that the toaster is plugged in and switched on.
2.
 - (a) Draw an AND–OR graph to show how to complete the goal ‘make tea’ shown in Figure 3.4.
 - (b) What changes would you need to make to the graph in Figure 3.4 to include the possibility of having coffee for breakfast?
3. Fred usually catches the 8.15 bus to school. He has a 10-minute walk to the bus stop. If he doesn’t have enough bus money, he can get a lift to school with Mr Brown next door, who leaves at 8.10 am. If he is late, he must walk!

Draw an AND–OR graph to show Fred’s different ways of getting to school.

Forward chaining

In forward-chaining expert systems, the system starts with facts supplied by the user and then draws conclusions from those facts.

In such a system, all the facts are held in the *working memory* and are constantly updated as the rules are *invoked*. The rules usually represent actions such as adding a new fact to the working memory and deleting an existing fact from the working memory. This approach is useful when you know all the initial facts but don't know what conclusions are likely.

In the breakfast example, we could represent the rules above as follows:

Rule 1 IF cereal is made
 AND toast is made
 AND tea is made
 THEN breakfast is complete.

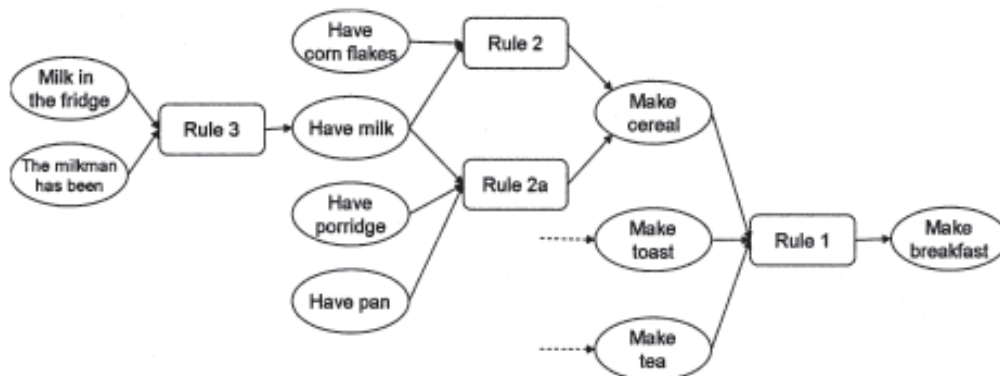
Rule 2 IF have cornflakes
 AND have milk
 THEN cereal is made.

Rule 2a IF have porridge
 AND have milk
 AND have pan
 THEN cereal is made.

Rule 3 IF milk in the fridge
 OR milkman has been
 THEN have milk.

These rules can be represented by the diagram in Figure 3.5.

Figure 3.5: Forward-chaining inferencing



Now suppose we start with the following set of facts in the working memory:

have cornflakes
have porridge
milk in the fridge
have pan

The first rule that can be applied (or 'fire') is rule 3, which adds a new fact, 'have milk', to the working memory, as follows:

have cornflakes
have porridge
milk in the fridge
have pan
have milk

Now there is a choice of which rule to fire: either of rules 2 and 2a may be used to continue the inferencing. There are a number of different methods that can be used to decide which rule should be used.⁴ These are discussed in Chapter 9.

Forward chaining starts with confirmed findings and then draws valid conclusions. For this reason, forward-chaining inferencing is known as *data-driven inferencing*.

Forward or backward chaining?

What type of inferencing method should be used for a given expert system? What are the advantages and disadvantages of each type of chaining?

The advantage of backward chaining is that it can lead quickly to a solution (if one exists). For this reason, backward chaining is well suited to problem solving where each part of the problem can be broken down into smaller sub-problems, until each sub-problem can be directly solved.

The main disadvantage of backward chaining is that it can take longer to find a solution if there are many hypotheses that must be tested and possibly discarded before finding the right one.

⁴ In a simple system, the rules will be used in the order they are entered. In this example, the result would be the same as for backward chaining.

The advantage of forward chaining is that it starts from the known facts, rather than suggesting a hypothesis and then trying to find the facts to support it. This means that it can lead more directly to a solution when the route to the solution is not predictable.

For example, when you go to the doctor, the first thing the doctor asks you is what is wrong, and you provide information about your symptoms, which starts the process of diagnosis. This is 'data driven' and uses forward chaining to find a diagnosis.

Imagine going to the doctor with a sprained ankle and the doctor asking these questions:

- Do you have a runny nose?
- Do you have a sore throat?
- Do you have sore ears?
- Do you have a high temperature?

The doctor might be asking these questions in an attempt to prove the hypotheses that you have a cold, a throat infection, an ear infection or flu! That would be the effect of using backward-chaining inferencing in medical diagnosis.

Forward chaining is used mainly in advice, classification and diagnosis systems, which are typically data-driven problems.

The disadvantage of forward chaining is that it is less predictable in the way that it leads to a solution because at any stage there may be many different rules that can be used to continue the inferencing. It is less easy to follow (or trace) the inferencing in a forward-chaining system than in a backward-chaining system.

More about forward and backward chaining

Forward and backward chaining are methods of *controlling the inferencing* and should not be confused with the *method of reasoning*. Although goal-driven reasoning is naturally suited to backward chaining, it is possible to implement a goal-driven reasoning strategy using forward chaining. Similarly, although data-driven reasoning is naturally suited to forward chaining, this strategy can be implemented using backward chaining. The MYCIN expert system discussed in Chapter 10 is an example of a data-driven backward-chaining system.

In general, any forward-chaining system can be implemented as a backward-chaining system, and vice versa. However, they are not simply inverses of each other and it is not generally possible to achieve this simply by rewriting the rules.

In practice, many expert systems use a combination of forward- and backward-chaining rules to obtain a solution. A good example is the INTERNIST medical expert system, which is described on page 111. Chapter 9 looks more closely at how forward-chaining systems work.

Exercise 3.2 (Int2)

1. At the scene of a crime, police detectives will look for clues and gather evidence. Once all the available evidence has been gathered, they can then draw valid conclusions. This is similar to inferencing in an expert system, but which form of inferencing: backward chaining or forward chaining? Give reasons for your answer.

Chapter 4: The user interface

The user interface in an expert system performs three main functions:

- it presents questions to the user and provides a means of inputting answers to questions
- it presents advice, conclusions and any other form of output resulting from the user's consultation
- it provides explanations to the user of how it arrived at its conclusion and why particular questions are being asked.

As with any piece of software, the user interface should be easy to use. However, in most expert systems, the design of the user interface is built into the shell and cannot be altered easily by the developers of the expert system.

Early expert systems were text-based systems, and a user consultation consisted of text-only output and keyboard input, as shown in Figure 4.1.

Figure 4.1: A consultation with the MIKE expert system shell

```
?- kb 'weather.pl' .
New knowledge base loaded.
-----
8 frame(s), 3 rule(s), 0 working memory element(s).
The current conflict resolution priority list (in order) is:
    [refractoriness, recency, specificity]
?- add 'the barometric pressure is rising' .
yes
?- add 'the western sky is cloudy' .
yes
?- deduce 'it is going to rain today' .
yes
?-
```

Input to an expert system

Current expert systems use a graphical user interface (as in Figures 4.2 and 4.3) or a web-based interface (as in Figure 4.4), and most operate using a question-and-answer mode of dialogue, as shown in the diagrams.

Figure 4.2: Input to the flex expert system shell

Figure 4.3: Input to the InterModeller expert system shell

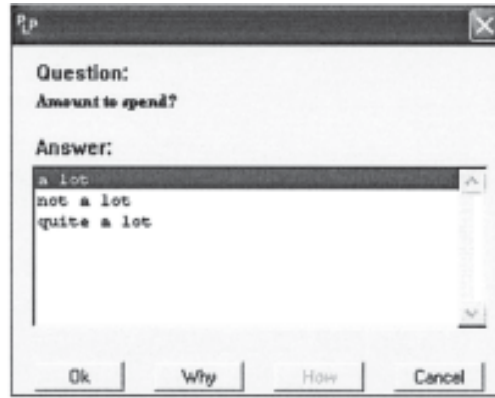
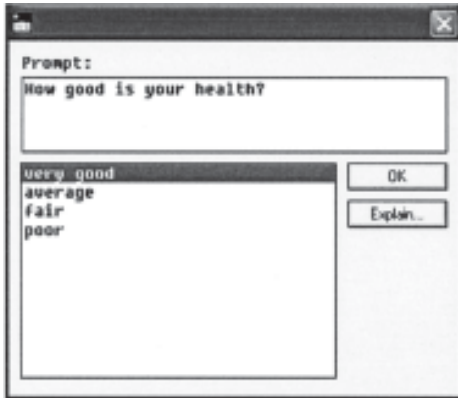
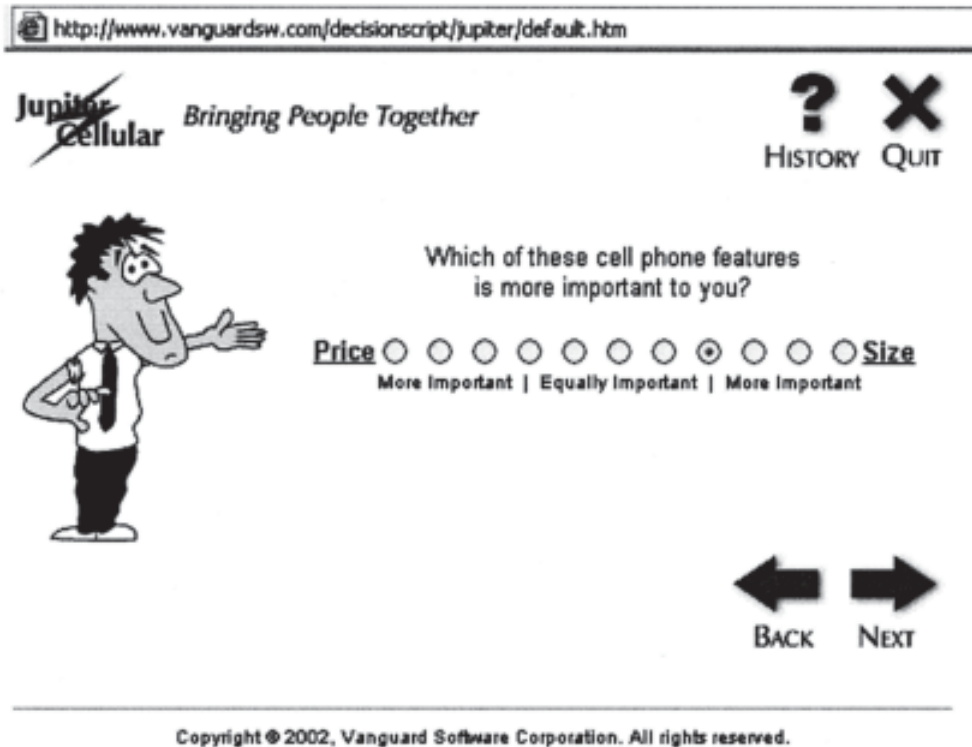


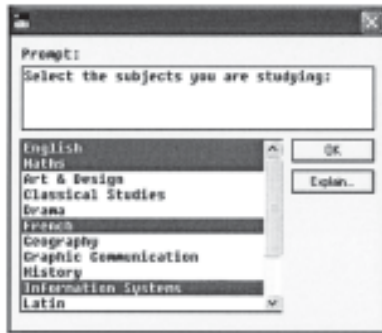
Figure 4.4: Input to the web-based DecisionScript expert system



As well as selecting a response from a list of choices or a set of options buttons, there are alternative methods of inputting data to an expert system:

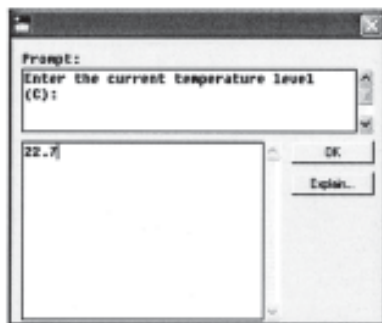
- selecting more than one response from a list (Figure 4.5)

Figure 4.5: Multiple-response input



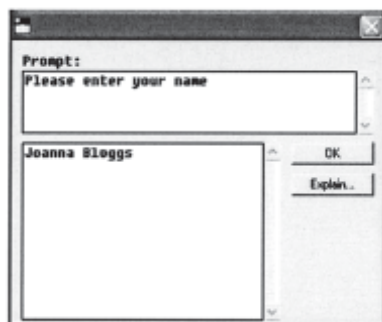
- input in the form of numerical value, e.g. a temperature reading from a thermometer (perhaps directly from a temperature sensor) (Figure 4.6)

Figure 4.6: Numerical input



- 'free text' input, e.g. for entering a user's name (Figure 4.7)

Figure 4.7: Free text input



- graphical input, e.g. to interpret scanned photographic or other images.

Output from an expert system

The output from an expert system is usually some kind of advice. This may be in the form of text (as in Figures 4.9 and 4.10), a table (as in Figures 4.11 and 1.3), or another form, such as a graph.

Figure 4.9: Output from the flex expert system shell

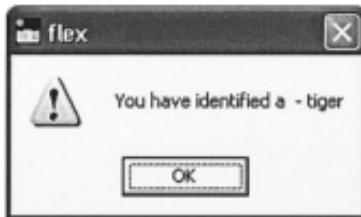


Figure 4.10: Output from the InterModeller expert system shell

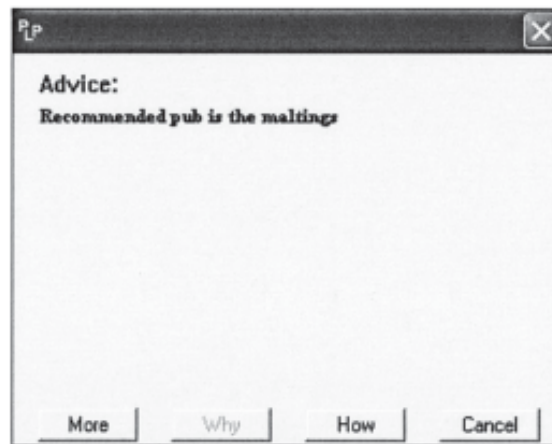





Figure 4.11: Output from the web-based DecisionScript expert system

http://www.vanguardsw.com/decisionscript/jupiter/default.htm

Jupiter Cellular Bringing People Together ? X
HISTORY QUIT

This table list the three phones that best match your preferences. Choose the phone you think is best for you by clicking on its image (our recommendation is at the top of the list). To see other options, click on *More Phones*.

Click on a column header to create a custom table sorted by that feature. Click again to reverse sort.

Score	Cell Phone	Image	Price	Dimensions	Talk	Standby
1	Nokia 5190		159.95	5.2x1.9x1.2	5	216
0.871	Mitsubishi G75		59.95	5.5x1.9x1.4	5	180
0.756	Ericsson 688		99.95	5.1x1.9x1.0	4	64

Explanations from an expert system

The explanation facilities of a user interface are required to explain the expert system's reasoning. These are also known as 'justification' facilities. There are two explanations that are required: why and how.

A 'Why' explanation is used to explain why a particular question is being asked. This can be in the form of a 'canned' explanation (a piece of stored text), as in Figure 4.13. A more sophisticated form of explanation justifies the question by explaining how the expert system can use the answer to the question to reach a conclusion, as in Figure 4.14.

Figure 4.13: 'Why' explanation of a question in the form of 'canned' text from the flex expert system shell

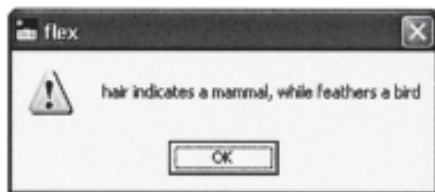
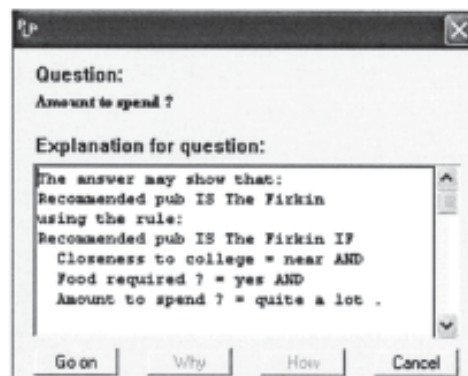


Figure 4.14: 'Why' explanation of a question from the InterModeller expert system shell



A 'How' explanation is used to explain how a particular conclusion has been reached. This involves displaying the user's responses to the questions given, as in Figure 4.15. A more sophisticated form of explanation justifies the conclusion in terms of the rules that have been applied, as in Figure 4.16.

Figure 4.15: Explanation of the advice given by the Decision expert system shell

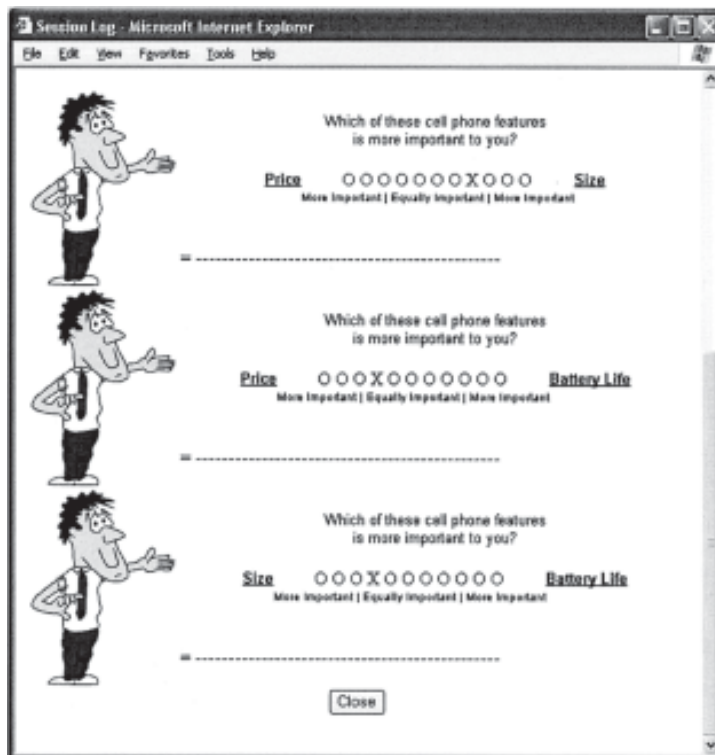
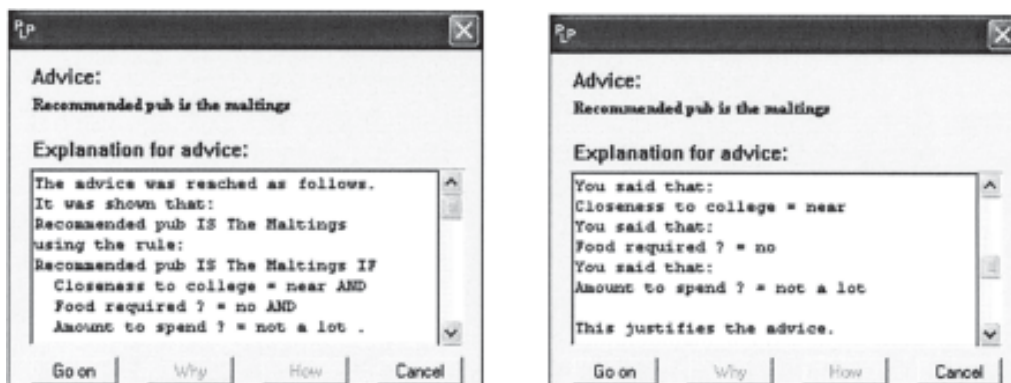


Figure 4.16: 'How' explanation of advice from the InterModeller expert system shell



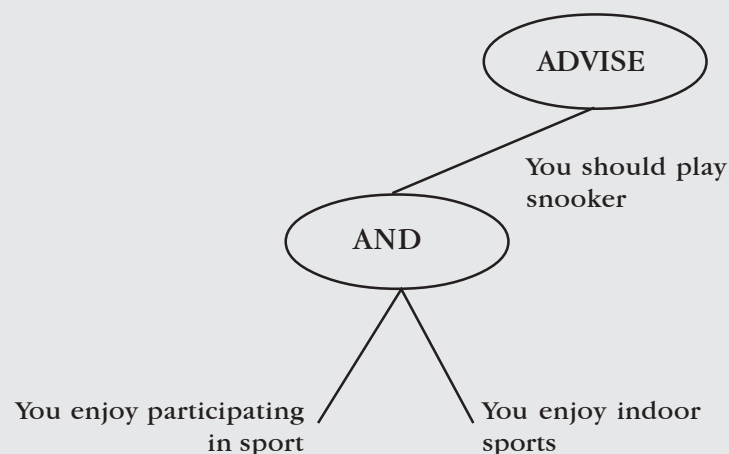
Generating explanations

To provide a more detailed explanation of the justification features of an expert system, we must consider the workings of the inference engine more closely. A *rule tree* provides a visual representation of the operations within the inference engine. The rule tree shows the knowledge contained in the knowledge base in the form of advice and conditions attached to the advice. At consultation time, inferencing can be compared with searching the rule tree for advice to give the user. The search method applied will determine the order in which the advice in the rule tree is encountered.

In a backward-chaining expert system, the process begins with a *hypothesis* (or piece of advice) at the bottom of the tree. The system then tries to confirm that hypothesis by seeking evidence to support it. It does this by working upwards through the tree testing each condition attached to the hypothesis. The conditions are tested by asking the user appropriate questions. If a user gives a negative response to a question, then the attached condition fails and the hypothesis is rejected. The expert system searches for the next hypothesis and the process continues.

When a user asks why a question is being asked, the expert system looks *up* its rule tree to find the hypothesis that it is trying to confirm. When a user asks how a conclusion was reached, the expert system looks *down* its rule tree to find the conditions attached to the hypothesis that has just been confirmed and then turned into advice.

Consider the rule tree below.



Some way during a consultation, the inference engine comes across the hypothesis: 'You should play snooker'. A sample dialogue is illustrated below.

Expert system: Do you enjoy participating in sport?
User: Yes.
Expert system: Do you enjoy indoor sports?
User: Why?
Expert system: Because I may be able to suggest that you should play snooker.
User: Yes.
Expert system: You should play snooker.
User: How?
Expert system: Because you said that you enjoy participating in sport and you said that you enjoy indoor sports.

The importance of explanations

It is important for an expert system to be able to explain its reasoning for a number of reasons. Explanations can:

- reassure the user that the expert system's conclusions are valid and are based on the evidence provided
- reassure the user that the questions being asked are relevant and necessary for drawing a conclusion
- be used to test an expert system to ensure that its conclusions are correct
- be used to debug an expert system if it is generating conclusions that are incorrect.

The issues of testing and debugging expert systems are part of the process of system validation, which is covered in Chapter 5.

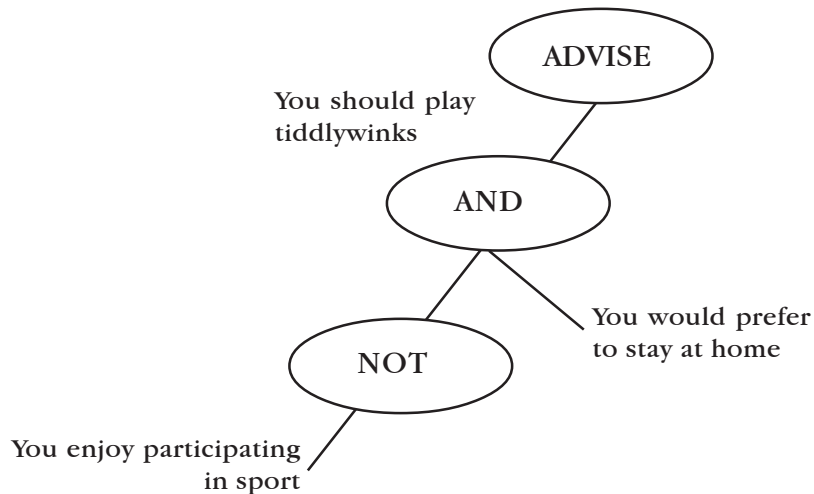
Exercise 4.1 (Int2)

1. State three ways in which the advice from an expert system can be presented.
2. An expert system can give the user two types of explanation of its reasoning. Describe both of these explanations.

Exercise 4.2 (H)

1. Why is the design of the user interface critical to the functionality of an expert system?
2. Describe how the explanation features of an expert system assist the system designer in:
 - (a) testing the expert system
 - (b) debugging the expert system.
3. Describe the advantages of an expert system's explanation features for:
 - (a) the users of a medical expert system
 - (b) the users of a financial planning expert system.
4. Describe the circumstances in which a user would want to make use of the:
 - (a) 'How' justification feature
 - (b) 'Why' justification feature.
5. Why is it important for an expert system to be able to provide justification for its actions?

6. Consider the rule tree below and the advice 'You should play tiddlywinks'.



- What hypothesis does the expert system adopt during consultation?
 - In trying to confirm this hypothesis, what is the first question asked by the expert system?
 - What justification does the expert system provide when the user then asks 'Why'?
 - Explain what would happen if the user responds with 'No'.
 - What difference would it have made if the user had responded with 'Yes'?
7. The following additional piece of advice about playing Twister is to be added to the knowledge base in question 6 above: *Twister is an indoor game – but it can be quite riotous and is only recommended for the sporty type.*
- Write down two appropriate conditions to attach to this advice.
 - Redraw the rule tree above to include this new knowledge.
 - During consultation, the new hypothesis 'You should play Twister' is tested. In order to confirm or reject this hypothesis, the expert system must ask questions. How many questions must the expert system ask if it is to prove this hypothesis?
 - Assuming that the user responds positively to all questions asked, what advice will be suggested by the expert system?
 - What response will the expert system provide when the user asks 'How'?

Chapter 5: Developing expert systems

Personnel

There are four main personnel involved in the development of expert systems:

- the domain expert
- the knowledge engineer
- the programmer
- the user.

Domain expert

The domain expert is the human whose knowledge and expertise has to be gained in order to design and construct the expert system.

Knowledge engineer

The knowledge engineer has the task of obtaining the domain knowledge from the human expert and representing it in a suitable form so that it can be entered into an expert system shell. This process is called *knowledge acquisition* and *elicitation*.

The knowledge engineer's role is similar to that of a systems analyst for other types of software development.

Programmer

The programmer's task is to enter facts and rules into the knowledge base from the knowledge representation provided by the knowledge engineer.

User

There are two categories of user: the 'client', and the 'end user'. The client may be a company that has commissioned the expert system and must be satisfied that it satisfies their requirements. The end user is the person who will ultimately use the expert system to help them solve problems. Both the client and the end user may be involved in testing the expert system to ensure that it works correctly.

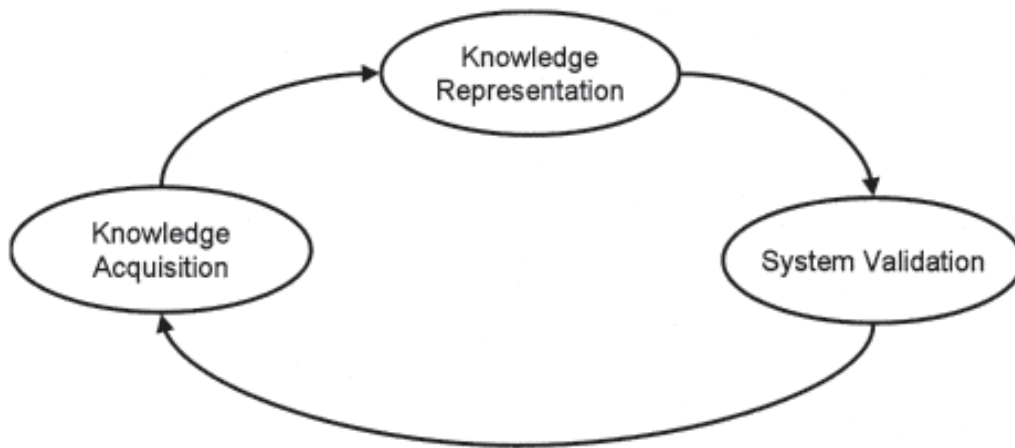
The expert system development cycle

Once the domain of an expert system has been identified, there are three stages involved in creating the expert system:

- **knowledge acquisition**
- **knowledge representation**
- **system validation.**

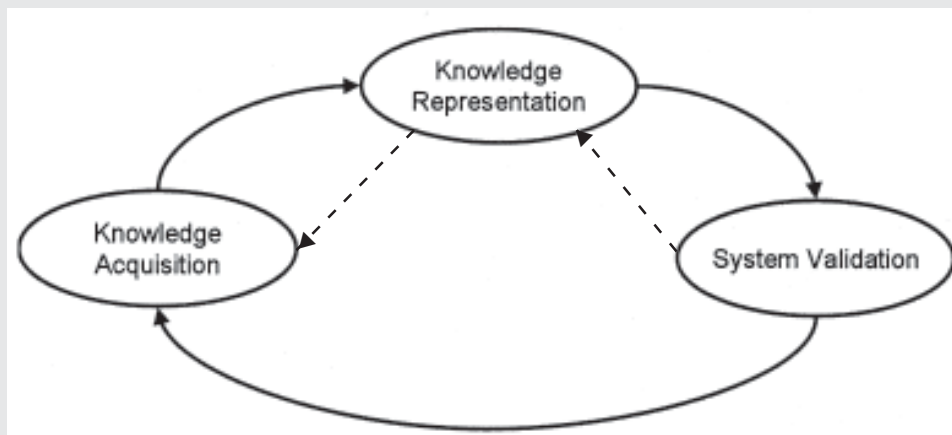
These stages are described below and are shown in Figure 5.1.

Figure 5.1: Diagram of the expert system development cycle



The stages of the development cycle are interdependent – for example, observations made during the system validation can lead to the knowledge acquisition stage being revisited. This then creates a cycle of events that together lead to the development of a fully working expert system.

Figure 5.2: Extended diagram of the expert system development cycle



Knowledge acquisition and elicitation

Knowledge acquisition (or elicitation) is the process of gathering together the domain knowledge required for the expert system's knowledge base. This knowledge may come from books, magazines, web-sites, CD-ROM reference materials or a human expert.

When information is obtained from books, magazines, web-sites and CD-ROM reference materials, it is called *knowledge acquisition*.

When information is obtained from a human expert, it is called *knowledge elicitation*.

It is important to remember that information displayed on the internet may not always be correct or accurate, and web-sites must be selected carefully to check they are reliable. While books, magazines and CD-ROMs may also sometimes contain incorrect information, they are usually checked by an editor before being published and so are more likely to be reliable.

More about knowledge acquisition and elicitation

The first step is to identify and select appropriate sources of information. This can be achieved as follows:

- Identification:** Who or what are the knowledge sources? Draw up a list of sources. This list may include books, other documents and on-line resources as well as human experts.
- Importance:** Rank the knowledge sources in order of priority of their importance
- Availability:** Rank the knowledge sources in order of availability. Books, other documents and on-line resources are much more easily available than human experts.
- Selection:** Select the sources based on their importance and availability.

The next step is to extract the appropriate knowledge from the selected sources.

If the source is text based (that is, books, other documents or on-line resources), then *knowledge acquisition* involves extracting the relevant material (perhaps by photocopying relevant pages and using a highlighter pen, or copying and pasting text from on-line resources into a word-processed document) and collating it to eliminate duplication.

If the source is a human expert, there are two main techniques that assist in the process of knowledge elicitation:

- Interview:** The knowledge engineer asks questions of the domain expert to explain how they solve problems and records a transcript of the interview.
- Observation:** The knowledge engineer observes the domain expert at work, asking questions as necessary to clarify the expert's reasoning and taking notes.

In both techniques, the recorded notes or transcript must later be analysed.

Issues in knowledge acquisition

The knowledge acquisition stage can often be very time-consuming and can lead to a bottleneck in the development process. One of the major problems with interviews and observation is that the knowledge engineer may well not understand the domain expert's jargon. Much of the knowledge engineer's time must be spent researching the knowledge domain to make sure that he or she has fully understood the issues involved. A knowledge engineer's field of expertise is in interviewing and extracting information from domain experts; he or she is not likely to be an expert, say, in the field of circuit board fault-finding or investment management. Another problem is that the expertise may be vague and imprecise, and can be very difficult to pin down. For example:

- '*in most cases*, I would prescribe the drug KN-42'
- 'if the soil is dry and crumbly it *usually indicates* that treatment is required'.

This imprecise knowledge is difficult for the knowledge engineer to deal with, but unfortunately expert knowledge is full of 'maybe', 'perhaps', 'probably' and 'doubtful'. The knowledge engineer must deal with each of these.

The reason for creating most expert systems is to enable them to be used by non-domain experts to solve problems that arise in the expert system's area of specialisation. For this reason, it is essential that the knowledge engineer not only extracts the specialist knowledge but also finds out how to apply that knowledge and use it intelligently. Without this understanding, the knowledge engineer will be unable to create an expert system that can solve problems.

One final difficulty in this stage is that the knowledge engineer has no way of dealing with common-sense knowledge in an expert system. There are occasions when the domain expert can't give a reason for a particular course of action because to him or her it is so obvious.

For example, most humans are experts at crossing the road. We stop at the kerbside, look for vehicles, wait until there are none and then cross the road. We would never dream of waiting 30 minutes at the kerbside for a parked car to move before crossing the road –

but an expert system would unless it was *specifically* told otherwise.

The knowledge engineer must try to ensure that every possibility has been covered and that nothing has been omitted simply because it was obvious to the domain expert. An expert system has no facility to enable it to deal with common-sense – each piece of knowledge contained in an expert system must be provided by a human expert.

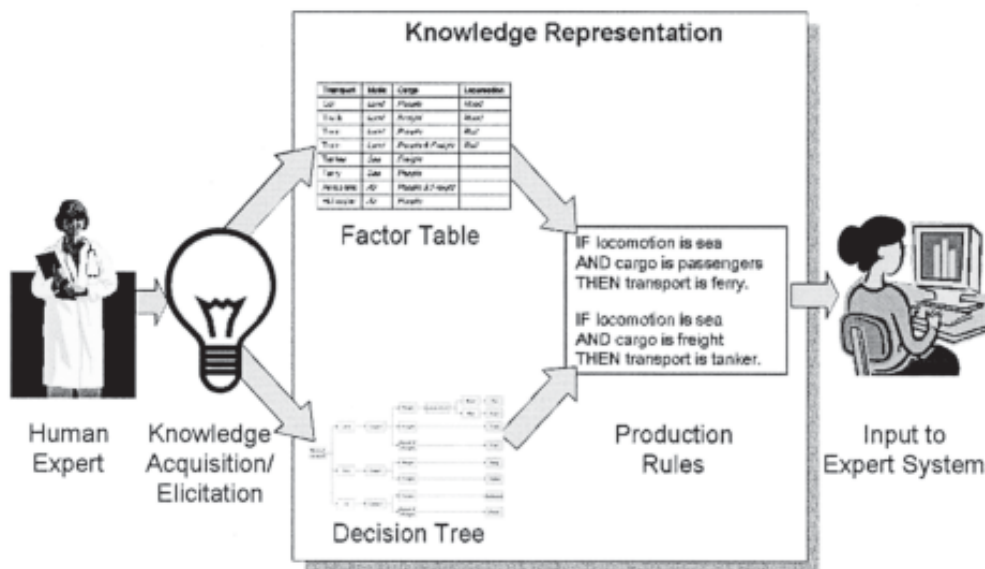
The goal of this stage of the development cycle is to extract some understanding of the chosen domain. To do so, you have to consult actual experts for that domain and learn a great deal about it yourself. Extracting some understanding of the domain from experts and literature is often a complex and yet vital task.

Knowledge representation

This stage in the development cycle involves representing the domain knowledge in a suitable knowledge representation language (KRL), ready for input to an expert system. There are a large number of techniques and tools that can be used for this purpose (in this situation, a tool is software that aids the creation of an expert system).

The knowledge engineer must extract the relevant information from the sources of expertise and organise it in such a way to make it suitable for use in an expert system. This will often be in the form of a factor table or decision tree, which will then be used to create a set of production rules that can be directly entered into a computer program or an expert system shell.

Figure 5.3: Knowledge representation in the expert system development cycle



More about knowledge representation

There are primarily three categories of KRL available to developers of expert systems:

- a 'traditional' third-generation programming language (3GL), e.g. BASIC, Pascal
- a 'symbolic' fourth-generation programming language (4GL), e.g. Prolog
- an expert system shell.

The main advantage of using a traditional 3GL is that this type of programming language is generally familiar to software developers. In addition, a visual programming language allows for the quick and easy creation of a user interface for the system. However, using a 3GL does not necessarily decrease the development time – indeed, it can often slow down the development process. This is because not only does the knowledge engineer have to represent the domain knowledge using the structures within the chosen language, but a method of inferencing must also be developed – all of which is very time-consuming.

Rather than develop the expert system in a 3GL, we could decide to use a symbolic 4GL, such as Prolog or LISP. These languages are superior tools for creating expert systems and other artificial intelligence (AI) software solutions. They allow knowledge to be represented as facts and rules, and they allow relationships between objects to be expressed. Prolog also has an in-built method of inferencing.⁵ By using one of the specialist AI languages, the development time is greatly reduced since there is no need to develop an inference engine. The knowledge engineer can concentrate instead on developing a suitable user interface and representing the domain knowledge.

Most expert systems, however, are developed using an expert system shell. An expert system shell is an 'empty' expert system – it is empty because it contains no knowledge – but it does have a user interface as well as an in-built method of inferencing. By using a shell, development time can be minimised and therefore costs can

⁵ Prolog's depth-first execution strategy corresponds directly to a backward-chaining inference strategy, but this can be successfully adapted to forward chaining.

be reduced. Shells allow expert systems for several different types of application to be developed since all that has to be created is the domain-specific knowledge base.

This means, however, that all knowledge bases created with the same shell will conform to the same set of in built rules. To all intents and purposes, they will look similar since they all have the same user interface and they will 'behave' in the same way since they all use the same method of inferencing. This can be a disadvantage of a shell.

In specialist knowledge representation languages, the method of inferencing is built-in and does not have to be programmed by the knowledge engineer. If an expert system is to be created using non-specialist languages, a vast amount of development time will be devoted to the design of the inference engine.

In selecting a KRL, it is best to make a shell your first choice as it will greatly reduce many of the development problems. If a shell is not available, or if the project does not fit a shell, then an AI language (4GL) such as Prolog should be your next choice. Only as a last resort should you use a traditional 3GL.

Having selected an appropriate development tool, the knowledge engineer then uses the structures available within the KRL to represent the domain knowledge.

System validation

This is often the most important stage in the development process. During this phase of development, the working expert system is tested and problems are identified. The method of testing used in expert systems is to compare the advice offered by the system with the advice offered by domain experts.

To enable a valid comparison to take place, the expert system is provided with the same information as the human expert – in other words, the same inputs are used. Only when the expert system consistently agrees with the domain experts is the system considered to be correct.

If the output from the expert system differs from that provided by human experts, then the problem must be identified. It could be that something has gone wrong at the knowledge representation stage – domain knowledge has been incorrectly represented, for example. Or it could be that there was a misunderstanding at the knowledge acquisition stage – perhaps the knowledge engineer misinterpreted information provided by the domain experts. Regardless of its source, it is vital that the problem is identified and resolved. Further testing will be required in order to validate any amendments made.

Exercise 5.1 (Int2)

1. Describe the three stages involved in creating an expert system.
2. Identify the four personnel involved in developing an expert system.
3. An expert system is a computer program. During the lifetime of a program, many people will interact with it. Which components of an expert system are the following people likely to interact with:
 - (a) a user seeking advice
 - (b) a software engineer entering facts and rules
 - (c) a user answering questions
 - (d) an expert system providing the user with an explanation of why a particular question was being asked?

Exercise 5.2 (H)

1. Describe each of the stages involved in creating an expert system.
2. What problems face the knowledge engineer during the knowledge acquisition phase of development?
3. An expert system that identifies and diagnoses problems with soil is being developed. During the knowledge representation phase, a suitable KRL must be selected. Describe the choices that are available, outlining the advantages and disadvantages of each.
4. Knowledge engineers prefer to develop expert systems using specialist knowledge representation languages. Why is this the case?
5. A finance company has commissioned a team of knowledge engineers to develop an expert system that provides investment advice. The team decide to develop the product using a shell. The company complains that the product is not unique – it is just the same as the expert system used by a rival building society. The knowledge engineers insist that the product is unique because it contains information about investment accounts that are unique to that finance company.

Who is correct – the finance company or the team of knowledge engineers? Explain your answer.

6. How are expert systems validated?
7. An expert system containing specialist knowledge of dinosaurs has been developed.

A recent dig in North America has uncovered new information about the bird-eating dinosaur *Ornitholestes*.

Explain what the software engineer must do to up-date the expert system with this new information.

8. A new expert system entitled 'Wild Flowers of Europe' is being developed.

How would the team responsible for developing this new expert system go about obtaining the knowledge required for inclusion in the knowledge base?

Chapter 6: Creating an expert system

A spare-time advisor

In this chapter, you will develop your own expert system to advise users on how best to spend their spare time. It will take into account factors such as cost, distance to travel and other participants, and give advice on activities to consider.

This exercise will enable you to learn what is involved in planning, creating, testing and evaluating an expert system.

Evidence of each of the following will be generated during the development of the expert system:

- statement of the purpose of the expert system
- identification of the domain of the expert system
- gathering of domain knowledge
- representing domain knowledge
- implementing the system
- testing the system.

Statement of the purpose of the expert system

The purpose of an expert system should be clearly stated and should identify the types of problems that can be solved using the expert system.

In this example, the purpose of the expert system is to provide the user with advice and suggestions on how to spend their leisure time. People would be able to use this expert system to get advice on what to do in their spare time.

Identification of the domain of the expert system

The extent of the information contained in an expert system is known as its domain. All expert systems have a very limited domain otherwise they would be too large and complex to deal with.

In this example, the domain of the expert system will be restricted to a certain number of rules based on the number of activities available. There will also be restrictions on the number of conditions that are taken into account in concluding a piece of advice.

Gathering domain knowledge

Before we can gather knowledge for an expert system, we must identify the source or sources of that knowledge. For most expert systems, the source is human experts in the particular domain or reference material from books.

For this expert system, we need to gather information about spare-time activities. We are likely to get the information we need by interviewing people with many spare-time interests.

Before we can interview, we must draw up a blank factor table to help us organise the information that we gather. The headings in the table should show the factors that affect each activity. In this case, suitable headings could be:

Activity	Cost: free, cheap, or expensive	Type of activity: sporting or non-sporting	Company: family, friends, or solo

Having created a suitable factor table, we would then proceed to complete the table with knowledge from our sources of expertise (e.g. domain experts).

Representing domain knowledge

Each row of the table corresponds to a rule that involves multiple conditions. For example:

```
ADVISE  Read a book
IF      you require an activity that is cheap
AND     you want a non-sporting activity
AND     you want a solo activity.
```

You should also note that the information provided in the last three columns of the table would lead to questions of a repetitive nature if the questioning were not structured carefully. It would therefore be appropriate to make use of the structured questioning within the KRL to

avoid this. For example, instead of repetitively questioning the user about the same thing:

Expert system: Do you want a solo activity?
User: No
Expert system: Do you want an activity with friends?
User: No
Expert system: Do you want an activity with family?
User: Yes

the knowledge base should be designed to allow a single question to be asked:

Expert system: What type of company do you want – solo, friends or family?
User: Family

Implementing the system

It is at this stage that we use the shell to enter the rules and create the expert system.

Internal documentation

When facts and rules are entered into the knowledge base of an expert system, it is important to lay out these out in a way that can be understood by someone else, who may be required to adapt or extend the expert system in the future.

Internal documentation is the way that the facts and rules are presented within the knowledge base. There are a number of golden rules to follow, as shown in Figure 6.1.

Figure 6.1: The golden rules of internal documentation

- Use sensible and meaningful names for facts, rules and values
- Make your rules as easily readable as possible
- Use spacing and indentation to layout your facts and rules so that the structure of each is clear
- Use comments to group parts of the knowledge base together, e.g. a set of related rules
- Use comments to identify the name, author, date and purpose of the expert system

Figure 6.2: Examples of good and poor internal documentation

```

% Species classification system
% A N Other
% October 2004
% Uses backward chaining

% Main rules
% Cheetah
relation species(cheetah)
if feeding_type(carnivore)
   and colour is tawny
   and marking is dark_spots
   and legs_and_neck is short.

% Tiger
relation species(tiger)
if feeding_type(carnivore)
   and colour is tawny
   and marking is black_stripes.

```

```

relation get(cheetah) if
  check(car) and col is
  tawny and mrk is dkspots
  and ln is short.
relation get(tiger) if check
  (car) and col
  is tawny and mrk
  is blstripes.
relation get(giraffe) if check
  (ung) and col is tawny and
  mrk is dkspots and ln is
  long.

```

Figure 6.2 shows examples of knowledge bases that are well documented and poorly documented. The first example is neatly laid out, includes comments, uses spacing and indentation to make the rules easy to understand, and uses sensible names for conditions and values in the rules.

The second example consists of the same rules, but, by contrast, is poorly laid out, includes no comments, and has a poor choice of names for conditions and values in the rules. As a result, it is much more difficult to make sense of!

You should ensure that you properly document any expert system you develop by following the rules listed in Figure 6.1.

Testing the system

It is necessary to test an expert system to make sure that it produces the correct advice for each set of possible inputs. To fully test the expert system, we will have to work our way through the conditions on each row of the interview table, checking that the advice suggested by the expert system matches the advice in the table. Since there are 20 pieces of advice in the completed table, we will have to run the expert system *at least* 20 times to make sure that it is correct.

Some expert system shells with a text-based interface have a 'trace' facility that can be used to record evidence of its correctness. Expert systems with graphical or web-based user interfaces may not have a trace facility. However, it may be possible to capture screen output of conclusions or use the explanation facilities (if these exist) to provide evidence of testing.

Once the expert system has been fully tested and has been found to be working correctly, we can produce a hard copy of the listing.

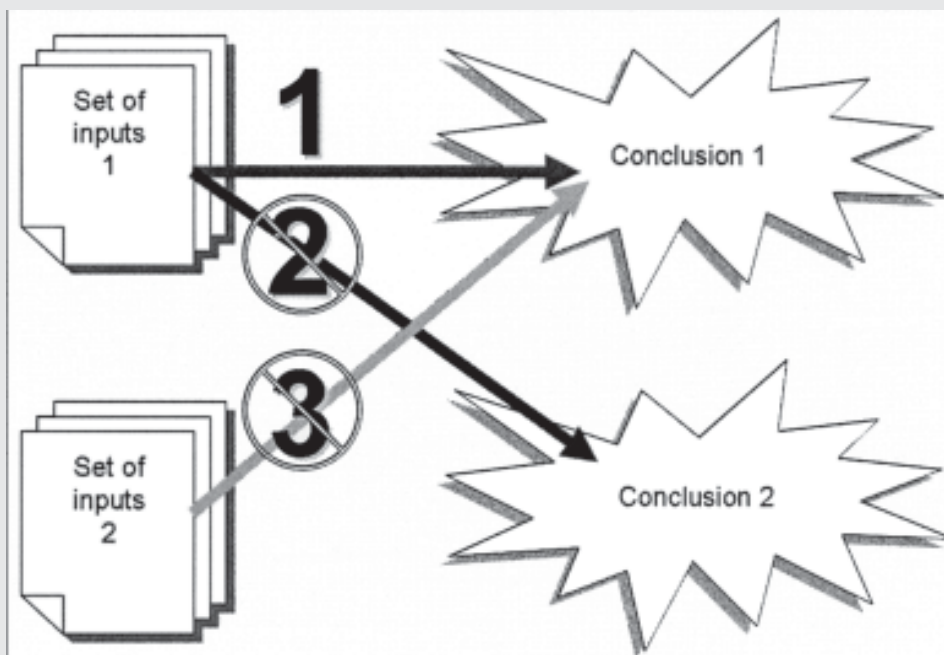
Structured testing

There are four categories of test that should be carried out:

1. testing that the correct conclusion is given for a set of inputs
2. testing that no incorrect conclusion can be given for a set of inputs
3. testing that no conclusion can be given for an incorrect set of inputs
4. testing that the system gives a sensible result if no conclusion is reached.

Tests 1, 2 and 3 are illustrated in Figure 6.3.

Figure 6.3: Diagram to illustrate testing strategies



Note that testing can never guarantee that an expert system is correct or 100% bug free. In particular, testing alone is not sufficient to guarantee conditions 2 and 3. However, structured testing can be carried out to satisfy 1 and 4, and this will help reduce the likelihood of problems arising in tests 2 and 3.

In large-scale expert systems there may be too many rules, and too many levels of chaining, to fully test every possible combination. In these situations, testing of individual rules and groups of rules must take place to ensure that these work as intended.

Exercise 6.1 (Int2)

1. An expert system that identifies vegetables is to be created using the knowledge contained in the factor table below.

Vegetable	Colour	Location of growth	Shape
Parsnip	White	Underground	Long and pointed
Cabbage	Green	Aboveground	Round
Carrot	Orange	Underground	Long and pointed

- (a) Use the structures available within your shell to represent the knowledge in the table as a set of rules.
 - (b) Enter these rules to create the knowledge base for your vegetable-identifying expert system.
2. Onions are round, white and grown below the ground whereas leeks are long green vegetables that are grown above the ground.

Write rules to represent this additional knowledge using the structures available within your shell. Enter these additional rules in your knowledge base.

3. Use your vegetable identifier to make sure that it correctly identifies:
 - (a) parsnips
 - (b) cabbages
 - (c) carrots
 - (d) onions
 - (e) leeks

Exercise 6.2 (Int2)

1. Draw up a factor table, like the one shown on page 32, to gather domain knowledge about 5–10 spare-time activities. You should obtain the information from others in your class or from your own interests and pursuits.
2. Enter rules into your expert system shell to correspond to the activities in your factor table. Pay particular attention to the structure of your rules to avoid unnecessary repetitive questioning, and to the layout of the knowledge base to ensure it is easily readable.
3. Add a comment to your knowledge base to indicate the purpose, author and date of the expert system. Add any other comments you feel are necessary.
4. Consult your expert system and make sure that you can correctly obtain advice about all the activities in your factor table.
5. Print out your knowledge base, together with evidence of your testing.

In Chapter 7 you will evaluate your expert system.

Exercise 6.3 (H)

1. Draw up a factor table, like the one shown on page 32, to gather domain knowledge about spare-time activities. You should extend the table with at least one further condition of your own.
2. At least one of the conditions in your factor table should be analysed further. For example, the condition 'cost is cheap' could be concluded if it is less than £3 and you have at least £3. You could further analyse the condition 'you have at least £3' by working out your income and expenditure this week!

You should analyse *one* condition to *two* levels of detail to illustrate the effects of chaining in your system.

3. You should obtain information about at least 15 activities from a range of sources, including books or magazines, the web, local advertising and human 'experts' (e.g. friends). You should list these, rank them in order of importance and availability, and then select the sources you will use.
4. Enter rules into your expert system shell to correspond to the activities in your factor table. Pay particular attention to the structure of your rules, to avoid unnecessary repetitive questioning, and to the layout of the knowledge base, to ensure it is easily readable.
5. Add a comment to your knowledge base to indicate the purpose, author and date of the expert system. Add any other comments you feel are necessary.
6. Consult your expert system and make sure that you can correctly obtain advice about all the activities in your factor table.
7. Test your system to ensure that it provides a sensible result if no activity can be concluded.
8. Print out your knowledge base, together with evidence of your testing.

In Chapter 7 you will evaluate your expert system.

Chapter 7: Evaluating expert systems

Why evaluate the system?

Having created an expert system, it is important to critically evaluate the system. This forms part of the system validation process.

This evaluation is essential for the maintenance and future enhancement of the system. It includes:

- purpose of the system
- range and coverage of rules
- quality of the user interface.

Purpose

This section describes the type, domain and purpose of the system.

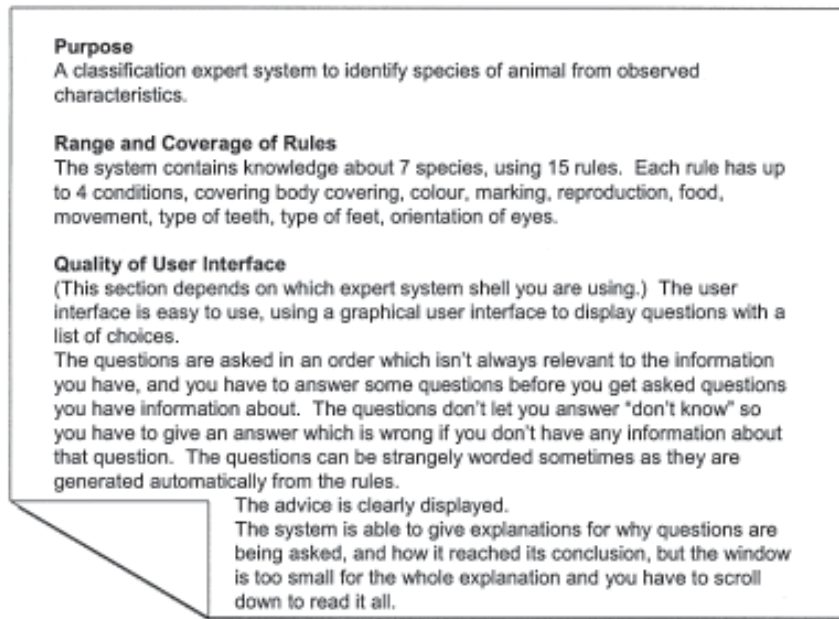
Range and coverage of rules

How many conclusions does this system contain? How many conditions or questions are included? How many rules are there? Which aspects of the domain are included and which excluded?

Quality of user interface

How good is the user interface? Is the system easy to use? Are the questions clear and well presented? Are the questions presented in an order that is logical? Is the conclusion clear and well presented? Is the system able to explain its reasoning ('How' and 'Why' explanations)? Figure 7.1 shows an example evaluation of the Species knowledge base:

Figure 7.1: Evaluation of the Species expert system



Detailed evaluation

A further measure of the quality of the interface relates to the structure of the questions.

Are the questions well-structured 'open' questions, with multiple choice answers, or are they 'closed' questions, which must all be answered yes or no?

Does the system allow the user to select more than one answer to a question, if appropriate?

How clear are the explanations given by the system?

More evaluation criteria

In addition to the items listed above, a more detailed evaluation will include:

- quality of reasoning
- correctness of conclusions
- fitness for purpose
- limitations of the system
- future enhancements.

Quality of reasoning

Does the system use forward or backward chaining? Is it goal driven or data driven? Does the method of reasoning suit the purpose of the expert system?

Correctness of conclusions

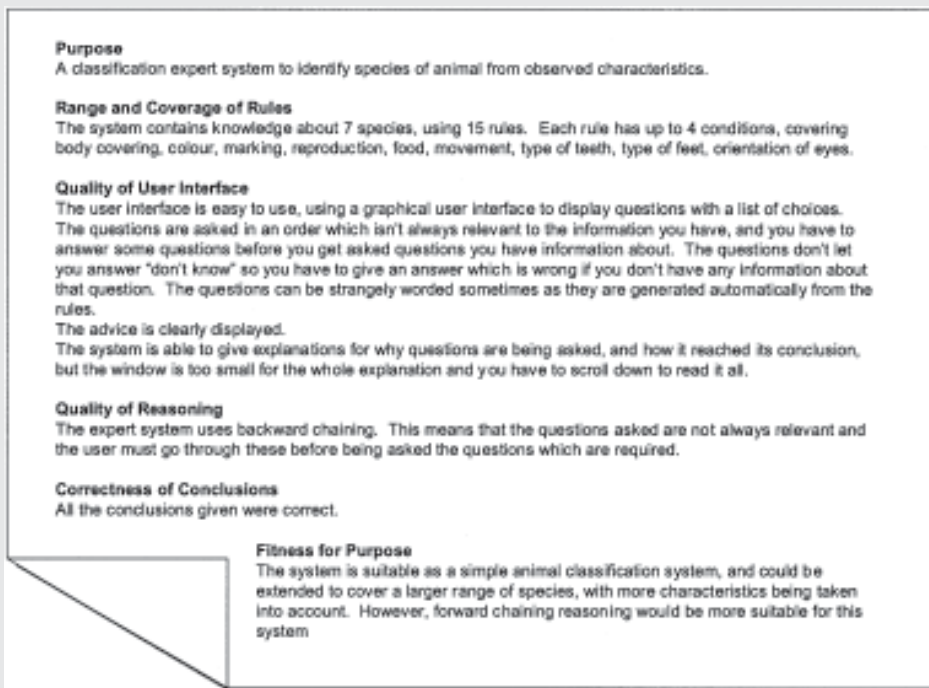
Are all the conclusions correct?

Fitness for purpose

Does the expert system succeed in what it was designed to achieve?

Figure 7.2 shows a more detailed evaluation of the Species expert system, taking into account these additional criteria.

Figure 7.2: Detailed evaluation of the Species expert system



Quality metrics

There are a number of other characteristics or 'metrics' that can be considered when evaluating an expert system:

- correct outputs given correct input
- complete output given correct input
- consistent output given the same input again
- reliable so that it does not 'crash' due to bugs (including stopping with no output)
- usable for people and user friendly
- validated to prove it satisfies the user's needs and requests
- tested for correctness and completeness
- cost-effective
- re-useable for other applications
- portable to other hardware/software platforms
- can be interfaced with other software, e.g. databases, web technology
- accurate and precise
- explanation facilities
- responds well at the 'limits of ignorance'
- understandable, well-commented code
- maintainable and enhanceable.

This list is useful for prioritising metrics as some of these may conflict. For example, full testing of the expert system for correctness and completeness will increase the cost of the system.

On the other hand, some metrics are interdependent. For example, well-commented code will increase the system's maintainability and enhanceability.

External documentation

Whereas internal documentation refers to the presentation of the knowledge base, external documentation refers to additional information that is produced about an expert system. This will include an evaluation of the system, but will also include details regarding the source(s) of the expert knowledge, transcripts of interviews and observation of experts, knowledge representation, testing strategy for system validation and any other relevant information (e.g. method of reasoning selected).

Exercise 7.1 (Int2)

1. Produce an evaluation of your spare-time advisor expert system, based on the headings listed on page 79.
2. Produce an evaluation of one of the expert systems used in the practical exercises.

Exercise 7.2 (H)

1. Produce a detailed evaluation of your spare-time advisor expert system based on the criteria listed on pages 79 and 80.
2. Produce a detailed evaluation of your extended Species expert system.

Chapters 8–11 are relevant for the Higher course only.

Chapter 8: Further knowledge representation

This chapter extends the knowledge representation techniques covered in Chapter 2 for candidates at Higher level. It may be helpful to recall the original passage of text:

There are three main modes of transport: land, sea and air.

Land-based locomotion is either by road or rail. Road-based locomotion includes cars, which are designed for carrying people, and trucks, which are designed for carrying freight. Trams and trains travel on rails, with trams designed for carrying people, while trains are designed for carrying people and freight.

Sea-based locomotion is by ship. Tankers are designed for carrying freight, while ferries are designed for carrying people.

Air travel is by aeroplane or helicopter. Aeroplanes are designed for carrying people and freight, while helicopters are designed for carrying people.

Decision trees

A *decision tree* is a graphical representation of the knowledge in a *factor table*.

Decision trees are used in a branch of artificial intelligence called machine learning, which attempts to produce programs that can learn from experience. This technique is of value in enabling expert systems to learn from consultations with users, add new facts and rules to their knowledge base to keep themselves up-to-date, and improve the accuracy of the advice they offer.

Figure 8.1 and 8.2 show examples of entering rules into an expert system shell using a decision tree.

Figure 8.1: This decision tree is used to work out the best form of transport for a journey (using the InterModeller expert system shell).

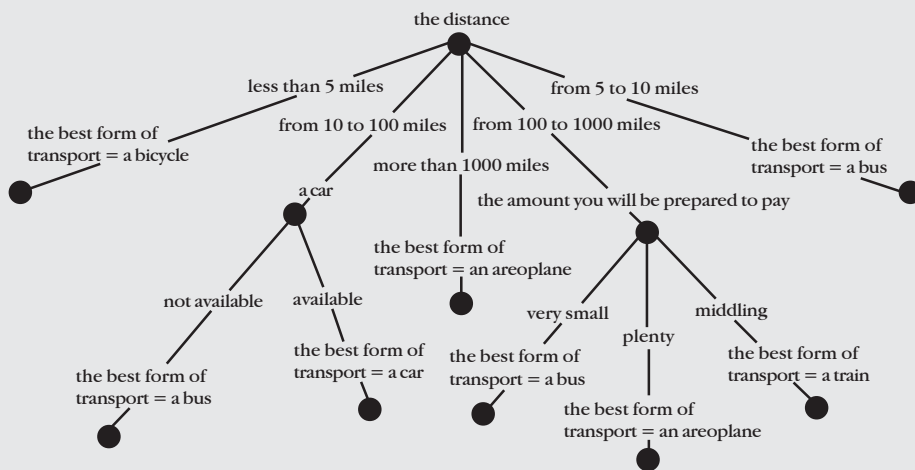
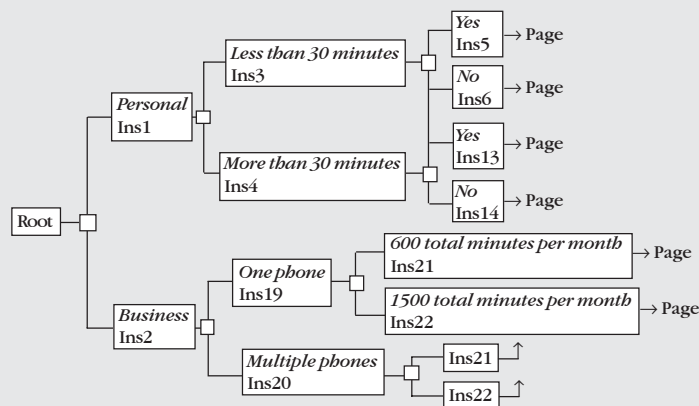


Figure 8.2: This decision tree is used to work out a user's mobile phone tariff (using the DecisionScript expert system shell).

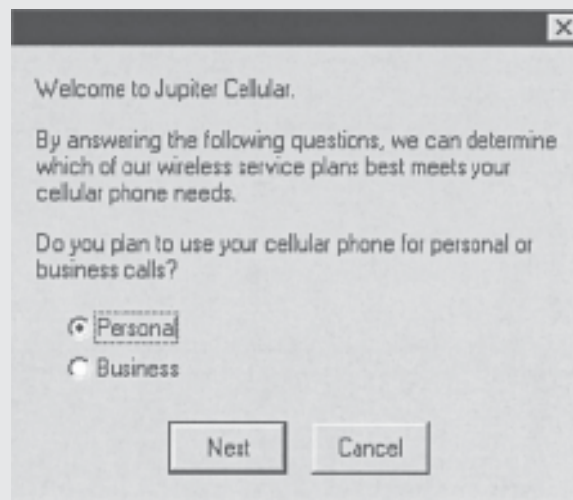


A decision tree may be drawn from top to bottom (as in Figure 8.1) or from left to right (as in Figure 8.2). Rather oddly, the 'top' of the tree is called the *root* (this is labelled in Figure 8.2), and at the 'bottom' of the tree are the *leaves*. The intermediate lines are called *branches*.

The 'junction points' in a decision tree represent *choice points* where a decision has to be made. The label of the choice point indicates the condition to be evaluated and hence the question to be asked (notice these are not shown in Figure 8.2). The branches that lead out from a choice point indicate the alternatives from which a choice must be selected.

Figure 8.3 shows the first question that might be asked by an expert system based on the decision tree in Figure 8.2.

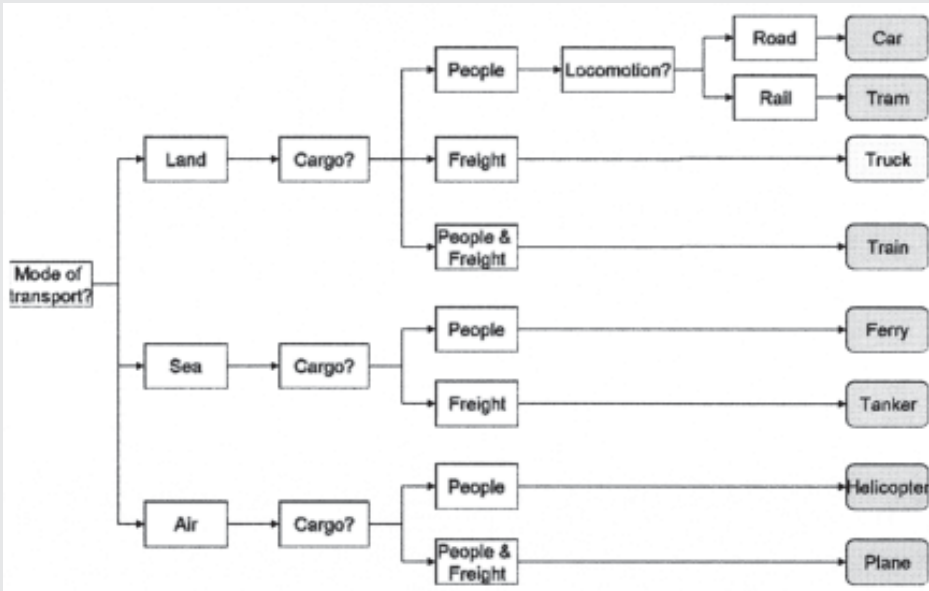
Figure 8.3: The first question based on the decision tree in Figure 8.2



The leaves of the decision tree indicate the conclusions that may be reached.

Here is the corresponding decision tree for the factor table on page 32.

Figure 8.4: Decision tree for the transport expert system in Chapter 2



Exercise 8.1

1. The factor table below shows information required to recommend a pub for college lecturers based on three factors – the distance from college, whether or not food is required and how much money is available to spend.

Draw a decision tree to represent the information in the factor table.

Closeness to college	Food required?	Amount to spend?	Recommended pub
Near	No	Not a lot	The Maltings
Very near	Yes	Not a lot	World's End
Near	Yes	Quite a lot	The Firkin
Quite far away	Yes		The Dome
Quite far away	Yes	A lot	Café Royal
Far away	Yes		Ryan's
Near	Yes	Not a lot	The Mitre

2. Draw a decision tree to represent the information in your completed factor table for the spare-time advisor expert system you created in Chapter 6.

Forward- and backward-chaining rules

As mentioned in Chapter 2, it is possible to rewrite production rules of the form:

```
IF <conditions>
THEN <actions>.
```

in the form:

```
<actions>
IF <conditions>.
```

While it is possible to do this for any production rule, the rules are interpreted in a different way by the inference engine. The rules are known as *backward-chaining rules* and *forward-chaining rules*.

A forward-chaining rule starts with the **conditions** and ends with the actions or **conclusions**. It has the general form:

```
IF <conditions>
THEN <actions>.
```

Often the <action> part of the rule has the effect of adding some new data to the working memory. Ultimately, there must be an action that displays some advice or conclusion to the user.

A backward-chaining rule starts with the goal to be satisfied and breaks this down into sub-goals. It has the general form:

```
<goal>
IF <sub-goals>.
```

Note: In general, any forward-chaining system can be implemented as a backward-chaining system, and vice versa. However, they are not simply inverses of each other and it is not generally possible to achieve this simply by rewriting the rules.

Logic

Expert systems attempt to simulate a human expert's reasoning capability. Reasoning is the application of logical thinking to solving a problem. Computers are good at performing logic (you will remember that part of a computer's central processing unit is the arithmetic and logic unit). All expert systems, therefore, use logical techniques – in fact, many are implemented using the AI programming language Prolog (= **Programming Logic**).

The next section is a simple introduction to using logic.

Propositional (zero-order) logic

Logic can be used to represent statements in English. These are called *propositions*. For example, consider the proposition 'It will rain today'. In logic, this can be represented using a kind of shorthand, or code, by giving this a letter, e.g. P:

so P = 'It will rain today'.

Now consider another proposition: 'The sky was red this morning'. This can be represented by the letter Q:

so Q = 'The sky was red this morning'.

We can use logic to represent the fact that 'if the sky was red this morning, then it will rain today'. (This is roughly equivalent to the well-known saying 'Red sky in the morning – shepherd's warning'.) This is done using the logical symbol \rightarrow , as follows:

$Q \rightarrow P$

This is read as 'Q implies P', or more simply 'If Q is true, then P is true'. The symbol \rightarrow is called the *implication operator*.

More complicated statements can be represented using propositional logic. Consider the statement 'If the sky was red this morning and the barometer reading is low then it will rain today'. This sentence contains three statements, the new one being 'the barometer reading is low'.

This could be represented using the code R, so R = 'The barometer reading is low'.

To connect statements Q and R together, we need to use the AND operator. The logic symbol for this is \wedge , so the sentence above would be represented in logic as

$$Q \wedge R \rightarrow P$$

This reads as 'Q and R implies P', or more simply 'If Q is true and R is true, then P is true'.

The logic symbol for OR is \vee , and the symbol for NOT is \neg . Consider the sentence 'If the sky was red this morning, and it is not after 12 noon, then it will rain today.' This could be represented as:

S = 'It is after 12 noon'

$$Q \wedge \neg S \rightarrow P$$

This reads as 'Q and not S implies P', or 'If Q is true and S is not true then P is true.'

The sentence 'If the sky was *not* red this morning or the sky was red last night then it will *not* rain today' could be represented as follows:

$$\neg Q \vee T \rightarrow \neg P$$

where T = 'The sky was red last night'.

Exercise 8.2

1. Let P = I will be late
Let Q = It is too far to walk home
Let S = I have £2 in my pocket
Let T = I will catch the bus

Represent the following expressions using propositional logic.

- (a) It is too far to walk home and I will be late.
 - (b) If it is too far to walk home and I have £2 in my pocket then I will catch the bus.
 - (c) If I do not have £2 in my pocket then I will not catch the bus.
 - (d) If it is too far to walk home or I do not have £2 in my pocket then I will be late.
2. In each of the following say whether or not the second sentence is an acceptable translation of the first. (If it isn't, think of a situation in which the sentences would have different truth values.) Hint: think about the first sentence being true (or false) and whether or not the second sentence would also be true (or false).
 - (a) No one is perfect. \neg Someone is perfect.
 - (b) Socrates is not wise. \neg Socrates is wise.
 - (c) Somebody isn't telling the truth. \neg Somebody is telling the truth.
 - (d) You may not look. \neg You may look.
 - (e) You must not look. \neg You must look.
 - (f) I don't think he is coming. \neg I think he is coming.
 3. In each of the following say whether or not the second sentence is a correct translation of the first. (If it isn't, think of a situation in which the sentences would have different truth values.)
 - (a) Mary and Jane are women. Mary is a woman \wedge Jane is a woman.
 - (b) Mary and Jane are friends. Mary is a friend \wedge Jane is a friend.
 - (c) Mary went to visit Jane, who lives next door.
Mary went to visit Jane \wedge Jane lives next door.

- (d) Mary went to visit someone who lives next door.
Mary went to visit someone \wedge someone lives next door.
- (e) Mary and John got married and had three children.
Mary and John got married \wedge Mary and John had three children.
- (f) Dumbo is a grey elephant. Dumbo is an elephant \wedge Dumbo is grey.
- (g) Dumbo is a small elephant. Dumbo is small \wedge Dumbo is an elephant.
- (h) Dumbo is an elephant, but quite intelligent.
Dumbo is an elephant \vee Dumbo is quite intelligent.
- (i) Dumbo is a skilful flier. Dumbo is skilful \wedge Dumbo is a flier.
4. In each of the following say whether or not the second sentence is a correct translation of the first. (If it isn't, think of a situation in which the sentences would have different truth values.)
- (a) Either he will or he won't. He will \vee he won't.
- (b) John will come today or tomorrow, but not both.
John will come today \vee John will come tomorrow.
- (c) You won't win unless you try. You won't win \vee you will try.
- (d) He is president of the United States or the most powerful man in the world.
He is president of the United States \vee he is the most powerful man in the world.
- (e) You may have soup or fish.
You may have soup \vee you may have fish.
- (f) John or David will come. John will come \vee David will come.
5. In each of the following say whether or not the second sentence is a correct translation of the first. (If it isn't, think of a situation in which the sentences would have different truth values.)
- (a) John was happy if Liverpool won the cup.
Liverpool won the cup \rightarrow John was happy.
- (b) John doesn't know if Liverpool won the cup.
Liverpool won the cup \rightarrow John doesn't know.
- (c) You won't succeed without trying.
 \neg You will try \rightarrow \neg You will succeed.

- (d) You will succeed only if you try.
You will try \rightarrow You will succeed.
- (e) You won't succeed unless you try.
 \neg You will try \rightarrow \neg You will succeed.
- (f) If my mother telephones, I shall be in the garden.
My mother will telephone \rightarrow I shall be in the garden.
- (g) If that egg has been boiled for 10 minutes, it is hard.
That egg has been boiled for 10 minutes \rightarrow That egg is hard.
- (h) If an egg has been boiled for 10 minutes, it is hard.
An egg has been boiled for 10 minutes \rightarrow An egg is hard.

Practical activity

If you have access to Prolog, you can try the following. Enter the following into Prolog's database,⁶ paying particular attention to capital letters, underscores and full stops:

```
the_sky_was_red_this_morning.
the_barometer_reading_is_low.
```

Note: In Prolog, all statements and rules use lower case letters and end in a full stop. The :- symbol corresponds to the \rightarrow operator and the comma corresponds to the \wedge AND operator.

At the query prompt ?-, enter the following queries:

```
?- the_sky_was_red_this_morning.
?- the_barometer_reading_is_low.
```

You should get the 'answer' yes to each of the queries.

Now try the following query:

```
?- it_will_rain_today.
```

What answer do you get?

⁶ In many versions of Prolog, you can do this by entering the following at the query prompt (?-):

```
?- consult(user).
```

When you have finished entering your facts and rules, finish the consultation by typing in:

```
?- seen. (or in some versions of Prolog, you can press <Escape>)
```

The problem is that Prolog has no way of connecting the facts it has in its database (or knowledge base) about the red sky this morning and the low barometer reading with the likelihood of rain. Instead, a rule must be supplied that makes this logical deduction.

Add the following rule to the Prolog database in the same way as before:

```
it_will_rain_today:-
    the_sky_was_red_this_morning,
    the_barometer_reading_is_low.
```

Now try the following query again:

```
?- it_will_rain_today.
```

What answer do you get this time?

Prolog has been able to draw its conclusion using the rule supplied together with the facts provided earlier.

Try entering a fact and a rule to represent the knowledge that it will *not* rain today, based on the saying 'Red sky at night – shepherd's delight'.

Predicate logic

While propositional logic is a simple way to represent knowledge, it has one important weakness, which is shown by the following example.

Consider these statements:

```
P = 'All men are mortal'
Q = 'Napoleon is a man'
R = 'Napoleon is mortal'
```

If P is true and Q is true, what should you be able to conclude about the truth of R? Using human logic it is easy to conclude that R is also true. This can be written in propositional logic using this rule:

$$P \wedge Q \rightarrow R$$

Now consider the statement $S = \text{'Socrates is a man'}$. Can we also conclude that Socrates is mortal?

Using human logic, the answer is 'yes' (by applying proposition P).

However, the statement 'Socrates is mortal' is a new proposition, and propositional logic requires a new rule to link this to the facts P and S, for example:

$T = \text{Socrates is mortal}$
 $P \wedge S \rightarrow T$

In fact, we would need an individual rule for everyone who has ever lived to be able to conclude that they are mortal!

Predicate logic allows us to write general rules to solve this problem. A predicate is a statement that uses brackets so that the contents of the brackets can change.

For example, in predicate logic, we could represent the statement 'Napoleon is a man' as the predicate:

(1) $\text{man}(\text{Napoleon})$

In the same way, we could represent the fact 'Napoleon is mortal' as:

(2) $\text{mortal}(\text{Napoleon})$

The predicates are numbered for easy reference. We can use the same *man* and *mortal* predicates to write statements about Socrates:

(3) $\text{man}(\text{Socrates})$
 (4) $\text{mortal}(\text{Socrates})$

In predicate logic, the statement 'all men are mortal' would be represented as follows:

(5) $\forall P: \text{man}(P) \rightarrow \text{mortal}(P)$

This reads as 'for all P, if P is a man, then P is mortal'.

The \forall symbol (a rotated A) stands for 'for all'.⁷

The statements $\text{man}(P)$ and $\text{mortal}(P)$ are called *predicates*. If P stands for Napoleon, and we know that Napoleon is a man, then we can conclude that Napoleon is mortal.

So, if we make $P = \text{Napoleon}$, then using predicate (5) we have a rule that says:

(6) $\text{man}(\text{Napoleon}) \rightarrow \text{mortal}(\text{Napoleon})$

This reads 'If Napoleon is a man, then Napoleon is mortal'. Using this rule with fact (1) $\text{man}(\text{Napoleon})$, we can draw the conclusion: $\text{mortal}(\text{Napoleon})$.

What makes this different from propositional logic is that it is no longer necessary to represent the statement that Napoleon is mortal because this can be concluded from rule (5).

The predicates:

(1) $\text{man}(\text{Napoleon})$

(5) $\forall P: \text{man}(P) \rightarrow \text{mortal}(P)$

allow us to conclude $\text{mortal}(\text{Napoleon})$.

This means we can use the predicate rule (5) to conclude the mortality of any person.

If P stands for Socrates, and we know that Socrates is a man, then we can also conclude that Socrates is mortal by applying fact (3) to rule (5):

(3) $\text{man}(\text{Socrates})$

(5) $\forall P: \text{man}(P) \rightarrow \text{mortal}(P)$

This is exactly how Prolog works and most expert systems work on the same principles.

⁷ Predicate logic uses the symbols \forall and \exists . The symbol \forall is called the *universal quantifier*. The rotated E stands for 'there exists', and is called the *existential quantifier*. Candidates are only required to use the universal quantifier in this unit.

Predicates can have two (or sometimes more) items inside the brackets. The predicate:

$\text{lives}(\text{Hamish, Scotland})$

can represent which country someone lives in. The same predicate can be used for different people and different countries, for example:

$\text{lives}(\text{Pierre, France})$

If the predicate

$\text{speaks}(\text{Hamish, English})$

represents which language someone speaks, then a general rule can be created to say which language is spoken by people living in each country, for example:

$\forall P: \text{lives}(P, \text{France}) \rightarrow \text{speaks}(P, \text{French})$

means 'all people who live in France speak French'.⁸

Exercise 8.3

1. Consider the following statements in predicate logic:

Statement	Meaning
$\text{plays}(\text{Sam, piano})$	Sam can play the piano.
$\text{likes}(\text{Susan, Sam})$	Susan likes Sam.

Express the following statements in predicate logic.

- (a) Amy can play violin and recorder.
- (b) If Sam can play the piano, then he can play the harpsichord.
- (c) Susan likes anyone who can play the bagpipes.

⁸ This is not strictly true, as some people in France speak other languages native to France, such as Breton or Basque, and immigrants to the country may also speak other languages.

2. Translate each of the following sentences into predicate logic, using the universal quantifier \forall .
- (a) All men are human.
 - (b) No women like John.
 - (c) Only women like John.
 - (d) Everyone is a man or a woman.
 - (e) If someone is a woman, she likes John.

Practical activity

If you have access to Prolog, you can try the following.

1. Enter the following to Prolog's database, paying particular attention to capital letters and full stops:
- ```
man(napoleon).
man(socrates).
woman(cleopatra).
woman(boadicea).
```
2. At the query prompt `?-`, enter the following queries. Write down the answer.
- (a) `?- man(napoleon).`
  - (b) `?- man(socrates).`
  - (c) `?- woman(cleopatra).`
  - (d) `?- woman(boadicea).`
- Prolog is able to answer 'yes' to the queries, because these facts are contained within the 'database' of facts.
3. Now add the following rule to the database:
- ```
mortal(P):-
    man(P).
```
4. At the query prompt `?-` enter the following queries. Write down the answer.
- (a) `?- mortal(napoleon).`
 - (b) `?- mortal(socrates).`
 - (c) `?- mortal(elvis).`

- (d) Explain the answer to (c).
- (e) ?- mortal(cleopatra).
- (f) Explain the answer to (e).

Prolog is able to answer 'yes' to queries (a) and (b), because it can use the rule entered to draw a logical deduction. However, the rule only applies to men, and then only to men it knows about. It is unable to draw any conclusions about men not included in the database. To draw deductions about the mortality of women, a similar rule must be provided.

5. Add the following rule to the database:

```
mortal(P):-  
    woman(P)
```

6. At the query prompt ?- enter the following queries. Write down the answer.

- (a) ?- mortal(napoleon).
- (b) ?- mortal(socrates).
- (c) ?- mortal(cleopatra).
- (d) ?- mortal(boadicea).

Chapter 9: Further inferencing strategies

This chapter extends the inferencing strategies covered in Chapter 3 for candidates at Higher level.

Chapter 3 introduced the two main inferencing strategies used in expert systems. It is now time to take a further look at each of these.

Forward chaining

Unlike backward chaining, where it is possible to trace exactly how the expert system is working, forward chaining is less easily predictable.

As we have seen, forward chaining works by maintaining a set of facts in the knowledge base, known as the *working memory*. Recall that a forward-chaining rule has the form:

IF <conditions>
THEN <actions>.

Each time a rule is invoked, or *fired*, the actions will typically result in a change being made to the contents of the working memory as new facts are added or existing facts are deleted. This process is called the 'select-execute cycle'.

The inference engine must then determine from the new state of the working memory, which rule or rules may next be fired. This is determined by comparing the facts in the working memory with the conditions listed in each rule. For a 'real-world' expert system of any scale, which may have thousands of rules, each with a large number of conditions, the task of matching up conditions to facts is a computationally expensive exercise and a special technique, called the RETE⁹ algorithm, is used to perform this task. The RETE algorithm is described later.

Conflict resolution

At each stage there may be several possible rules that may be fired. The inference engine must decide which of the rules should be selected. The set of possible rules is called a *conflict set*. There are

⁹ pronounced 'retty'

a number of *conflict resolution* techniques that can be used to select a rule to be fired. These are described in turn below.

Rule ordering (first-come-first-served)

The most straightforward method of conflict resolution is to simply select the rule from the conflict set that appears first in the knowledge base. This means that the order in which rules are entered into the knowledge base is important and must be carefully thought out.

Recency

The idea behind this method is that the inference engine should attempt to 'follow a line of enquiry' by firing rules that make use of the data that have been most recently added to the knowledge base. Rules that use more recent data are selected in preference to others that use data that have been in the working memory for some time. Of course, this method may fail to reach a conclusion and the inference engine may have to revert to using rules that use older data in order to make progress.

Specificity (size ordering)

Using this method, the inference engine selects rules with the largest number of conditions (that is, are the most specific) in preference to more general rules with fewer conditions. The idea is to attempt to focus the search for a solution as narrowly as possible. Again, of course, it may be that the focus has to be widened at some point, in which case the inference engine may have to revert to using more general rules.

Refractoriness

This rather clumsy word simply means 'don't fire the same rule more than once'. This comes in two forms: the first is to remove a rule from the conflict set once it has fired so that it can't be fired twice in succession; the second is to remove the rule from the conflict set permanently so that it can't be fired again (with the same data). Either way, the idea is to avoid the inference engine going round in circles, or looping.

Many systems use recency, specificity and refractoriness together to create a conflict resolution strategy. However, there are two further methods to consider, which may be used in addition to, or instead of, those just described.

Data ordering

Not all things are created equal – some things are just more important than others – and that applies to information as well. The idea behind this method is that the inference engine should select rules that use more important data in preference to rules that use less important data. Of course, as with the other methods, it may be that what was thought to be important turns out not to be – as ever, things may not always be as they seem! In this case, the inference engine must revert to using the less important rules.

Context limiting (setting a rule agenda)

The idea here is that different sets of rules may be designed for different sets of circumstances and should therefore only be considered as a set on their own. For example, imagine a cinema expert system for advising on which film to see at which cinema. Some rules are concerned with the choice of film, others with the choice of cinema to go to. It doesn't make sense to check the cinema rules for firing until a film has been selected, and once a film has been selected the film rules can then be disregarded.

This process is called *context limiting* and the set of rules being considered at any one time is called the *rule agenda*.

Finding a conflict set: the RETE algorithm

As mentioned above, at each stage in its select–execute cycle, the inference engine must compare the facts in the working memory with the conditions listed in each rule in order to determine which rules may be fired. Because the number of comparisons required to match up conditions to facts may be very large, the RETE algorithm has been developed to do this in an efficient manner.

RETE is based on two observations about how forward chaining affects the working memory:

- when each rule is fired, its actions will only change a few facts in the working memory
- many rules share conditions in common: that is, the same condition may appear with more than one rule.

These observations allow the RETE algorithm to store the current state of matches between facts and conditions in rules. Each time a fact is added or deleted, it is only necessary to look at those rules with conditions relating to the change in the working memory.

Handling uncertainty

Back in Chapter 1, the following definition of an expert system was introduced:

An expert system is a computer program that:

- contains the specialist knowledge of one or more human experts; this expert knowledge is in a form that others may use to solve problems in a specific domain
- provides the user with advice via a consultation
- can explain the advice it gives and why it is asking particular questions.

We can add the following to this definition:

- an expert system must be able to handle uncertain and incomplete information.

There can be several different reasons why information may be uncertain or incomplete:

- (a) Domain experts may not be 100% certain of their knowledge.
 - Ecologist: 'Global warming may be the cause of rising sea levels'.
 - Doctor: 'Lambda39 is likely to be the most effective drug to prescribe if the infection is of the Kamon-beta strain'.

(b) Information about the problem to be solved may be incomplete or unreliable.

- Car driver: 'Perhaps there was oil on the road'.
- Patient: 'It is a kind of dull aching pain'.
- Student: 'I managed to do some studying'.

For example, when consulting a medical expert system, we may not be quite sure that some symptom is present in the patient; we may not be absolutely sure that some measurement data is accurate. It is also possible that the human domain experts who provided the knowledge contained in the medical expert system were unsure of their facts – 'this drug may cause problems but usually does not'.

Certainty factors

There are a number of methods of enabling expert systems to handle such information. The simplest of these is to use *certainty factors*.

A *certainty factor* is a number that is attached to a rule to indicate the degree of confidence in the conclusion of the rule, given that its conditions are known to be true. The certainty factor is a percentage, usually expressed as a whole number between 0 and 100 (although it may also be expressed as a decimal number between 0 and 1). For example, here is a rule predicting the likelihood of rain given the colour of the sky this morning:

```
IF the sky was red this morning
THEN it will rain today
CF 40.
```

This rule is based on the saying 'Red sky in the morning – shepherd's warning', and says that if it is true that the sky was red this morning, then there is a 40% chance that it will rain today. In this case, the figure 40 indicates the degree of certainty a domain expert has in this rule.

However, what degree of certainty is there that it will rain today if it is uncertain whether the sky was red this morning or how red it was? If it is less than 100% certain, then the certainty of rain today – based on this rule – must be less than 40%. Suppose it is only

80% certain that the sky was red this morning, perhaps because sky wasn't very red or because there is some doubt in recalling the fact.

The certainty of rain today is calculated by multiplying the certainty factor of the condition by the certainty factor of the rule, that is:

$$CF_{\text{condition}} \times CF_{\text{rule}}$$

$$\frac{80}{100} \times 40 = 32$$

The certainty of rain today, based on this rule, is 32%.

Using certainty factors with more than one condition

Now consider this more sophisticated rule for predicting rain:

IF the sky was red this morning
AND the barometer is low
THEN it will rain today
CF 70.

Now suppose 80% certainty that the sky was red this morning, as before, and 60% certainty that the barometer was low, perhaps because of some uncertainty as to how low the reading was. What then is the certainty of rain today?

It is clearly less than 70%, but how much less?

Applying the formula above with the certainty factor of a red sky would give a result of 56%:

$$CF_{\text{condition}} \times CF_{\text{rule}} = \frac{80}{100} \times 70 = 56$$

However, a calculation based on the certainty of the barometer reading would give the result 42%:

$$CF_{\text{condition}} \times CF_{\text{rule}} = \frac{60}{100} \times 70 = 42$$

In reality, the answer is somewhere in between.

A simple, although not very accurate, solution is to say that it is

certainly no less than 42%, and to give this as a conservative estimate. This is the method used by the MYCIN medical expert system and by the InterModeller expert system shell. It can be summarised by the formula:

$$CF_{\text{condition}} = \min(CF_{\text{cond1}}, CF_{\text{cond2}}, \dots, CF_{\text{condn}}) \times CF_{\text{rule}}$$

In plain language this means 'take the lowest certainty factor of all the conditions and multiply by the certainty factor of the rule'.

There are many more sophisticated mathematical models of probability that can be applied in expert systems. These are not required for this unit.

Exercise 9.1 (H)

1. Which word(s) in each of the examples given in (a) and (b) on pages 99 and 100 indicate uncertainty in the information being provided?
2. Identify five more words or phrases that can indicate uncertainty.
3. Give two reasons why expert systems may have to handle uncertain or incomplete information.
4. Consider an expert system containing the following forward-chaining rules:
 - 1 If A and D then E
 - 2 If A and B then G
 - 3 If A and C then E
 - 4 If A and E then F
 - 5 If A and B and C then D
 - 6 If A and C and D then H

Suppose the working memory contains the facts A, B and C, added in the order listed.

- (a) Which rules exist in the conflict set?
 - (b) Which rule will fire using a *first-come-first-served* (also known as *rule ordering*) conflict resolution strategy?
 - (c) Which rule will fire using the recency conflict resolution strategy?
 - (d) Which rule will fire using the specificity conflict resolution strategy?
 - (e) After a rule has fired, what is the effect of the refractoriness strategy?
5. Describe three uses of certainty factors in an expert system.
 6. An insurance company uses an expert system to assist in processing insurance claims. A claim is received from the driver of a car involved in a collision.

Write down two examples of uncertain or incomplete information that the driver may provide in this situation.

Chapter 10: Classical expert systems

This chapter looks at some of the most important expert systems that have been developed and discusses the contributions each of them has made to the development of expert systems technology.

MYCIN

Although not the earliest expert system, MYCIN is certainly one of the most famous. It was a medical expert system, developed in 1972, and its domain was bacterial blood infections. Its purpose was to classify blood samples in order to assist doctors who were not experts in the treatment of blood infections to prescribe suitable antibiotics.

Put simply, the idea is this. A patient with a blood infection is very seriously ill and must be treated quickly. However, to provide a treatment, you need to know what the source of the infection is. This requires a blood sample to be taken and analysed. The blood sample is sent to the laboratory where a culture of the infecting organism grown. Unfortunately this takes around 48 hours, and if doctors waited until this was complete their patient might be dead!

So, doctors have to come up with quick guesses about likely problems from the available data and use these guesses to provide a temporary treatment where drugs are given that should deal with any possible problems that might develop until the results are obtained from the laboratory and full treatment can be provided.

MYCIN represented its knowledge as a set of production rules, of the form IF <conditions> THEN <conclusion>. It was a data-driven system, which worked by breaking the problem down into smaller sub-problems, each of which could be solved using backward chaining to derive its conclusions. It was able to provide a number of alternative analyses for the blood sample. It also introduced the idea of certainty factors to allow for the fact that the analysis could never produce results that were 100% certain, and to rank the alternative solutions. It represented this uncertainty as a decimal number between 0 and 1. Here is an example of a MYCIN production rule:

- IF (1) The stain of the organism is Gram negative, AND
 (2) The morphology of the organism is rod, AND
 (3) The aerobicity of the organism is aerobic
 THEN There is strongly suggestive evidence (0.8) that the
 class of organism is Enterobacteriaceae

DENDRAL

DENDRAL was the earliest expert system to be developed, in 1965, and was the first computer program to show that it was possible to rival the performance of human experts in a specialised field.

Its domain was molecular chemistry and its purpose was to classify unknown organic compounds, determining their molecular structure by analysing and interpreting data from a device called a mass spectrometer.

Put simply, the idea is this. To find out what kind of chemical compound you have, you analyse it with a mass spectrometer. This works by splitting the chemical up into pieces and you have to work out from the size (or mass) of the pieces what the original compound is. It is a difficult problem to solve because there are a very large number of compounds, and each can be split up in a number of different ways. That means that any piece of a given mass could belong to any of a number of different compounds. Worse still, different pieces can have the same mass, so you can't even tell from the mass what piece of a compound you have!

To illustrate, there are around 10 000 compounds (called isomers) that have the chemical formula $C_6H_{13}NO_2$. However, the actual structure of the compound can be determined by applying some constraints based on facts that are already known about the compound – for example 'the compound must be like this' or 'the compound cannot be like that'.

DENDRAL was able to succeed in its objective by applying constraints like these to reduce the number of possibilities.

ONCOCIN

ONCOCIN, as its name suggests, was similar to MYCIN, in that its domain was medical, in this case related to the field of oncology (cancer). However, it was a planning system, and its purpose was to devise a course of cancer treatment for a patient. It was developed in 1981.

Its main contribution to the development of expert systems was the way in which it acquired its knowledge from the human domain experts. Most expert systems obtained their domain knowledge by direct entry of facts and rules into the knowledge base. ONCOCIN attempted to automate the process of knowledge acquisition by consulting human domain experts directly, using a sort of interview session at the computer terminal. The system provided a set of screen-based forms for the human expert to fill in with details of drugs, dosages and blood test results. This process was extended further to provide a graphical user interface that used icons to represent the sequences of treatments that patients should follow. These inputs were then translated automatically into production rules, which were entered into the knowledge base.

The main advantage of this method of acquiring domain knowledge is that it reduced the time and cost involved in maintaining the knowledge base, and meant that there was no requirement for a knowledge engineer and programmer to add new domain knowledge.

R1/XCON

The R1 expert system was developed in the early 1980s and was one of the most successful systems ever produced. Its commercial version was called XCON. It was a planning system and its domain was the configuration of VAX computer systems.

To understand why it was so successful, we have to understand what computer systems were like in the 1970s and 1980s. Mass produced personal computers as we know them today were unknown in the 1970s. The first IBM PC was only introduced in 1981 and the first computer to use a mouse and a graphical user interface was the Apple Macintosh in 1984. The Windows operating system for IBM-compatible PCs was only introduced in large numbers in 1991 (Windows 3.1). E-mail and internet usage became common only after the introduction of the world wide web in 1993. Networked systems were unheard of – mainframes and minis (smaller mainframe-type systems) were the order of the day. These computer systems were large and would have easily filled a classroom! Very few components were internal – disk drives or tape streamers were large, unwieldy pieces of equipment that were the size of free-standing fridges!

Organisations that required computers typically had to have such mainframe and mini systems custom-built to order. Configuration of the components was a complicated process. The system was provided with information about how many users were required, how much memory was required, and so on, and would provide floor plans, engines, boards and even correct cable lengths.

XCON did in seconds what people could do in minutes or hours. As a measure of the scale of a real-world expert system, by 1983, XCON had over 3000 rules.

R1/XCON was implemented using the OPS5 planning system (see below). Like MYCIN, it used production rules to represent knowledge. Here is an example of an R1 rule:

DISTRIBUTE-MB-DEVICES-3

IF the most current active context is distributing massbus devices
 & there is a single-port disk drive that has not been assigned to a massbus
 & there are no unassigned dual-port disk drives
 & the number of devices that each massbus should support is known
 & there is a massbus that has been assigned to at least one disk drive and that should support additional disk drives
 & the type of cable needed to connect the disk drive to the previous device on the disk drive is known
 THEN assign the disk drive to the massbus.

At each stage of the configuration process, R1 would have many rules that could be applied. The main conflict resolution strategy used was *specificity* – R1 would attempt to apply special case rules before more general rules.

However, unlike MYCIN, R1 was a data-driven system that derived its conclusion using forward chaining.

INTERNIST

INTERNIST is a diagnosis system and its domain is medical diagnosis.

Its purpose is to model (or imitate) the process of diagnosis used by human medical experts.

This is the way a consultation with a doctor usually results in a diagnosis. The patient has a number of symptoms, which will lead

the doctor to suspect a number of possible causes (i.e. hypotheses). As a result, the doctor will then expect certain other symptoms to be present or absent, and will carry out some observations or tests to confirm or discard these. These observations and tests may lead to further hypotheses being considered as to what is wrong with the patient.

The initial consultation with the doctor is forward chaining and data driven. That is, the patient provides the data (i.e. what is wrong with them) and the doctor works forward from there to try to find a diagnosis. However, in practice, there are many diseases that could be diagnosed from a set of symptoms, and the same disease could manifest itself in different ways with different sets of symptoms. As a measure of the scale of a real-world expert system, INTERNIST contains around 100,000 relationships between symptoms and diseases.

A simple application of forward chaining is not going to lead to a satisfactory conclusion by itself. Instead, INTERNIST combines forward chaining with backward chaining to try to find further evidence to support or rule out possible diagnoses.

PROSPECTOR

PROSPECTOR is an advice system based on the domain of mineral geology. Its purpose was to aid geologists in their search for mineral ore deposits.

Given field data about a geological region, PROSPECTOR can determine the probability of discovering a range of ore deposits. Its expertise is based on geological rules that form models of ore deposits and a database of known rocks and minerals.

In 1980, PROSPECTOR analysed geological data from a site near Mount Tolman in eastern Washington State in the USA and predicted the existence of molybdenum (a mineral ore) in a particular location. Subsequent drilling by a mining company confirmed PROSPECTOR's prediction.

PROSPECTOR is important because it used a form of inferencing called *fuzzy logic* to derive its conclusions. Like MYCIN, PROSPECTOR had to work with evidence that was incomplete or uncertain. However, whereas MYCIN used the more ad hoc approach of certainty factors as a measure of confidence in its conclusions, PROSPECTOR used *probabilistic* reasoning, which used the mathematical theories of

probability to derive its conclusions. The conclusions drawn by PROSPECTOR match those of the human expert to a very high degree of accuracy.

OPS5

OPS5 is a knowledge representation language designed for representing knowledge for use in *production systems*. A production system is one that uses forward chaining to add new facts by applying rules to existing facts.

OPS5 is implemented using the LISP programming language. Here is a fact in OPS5:

```
(Student ^ Name Joe Bloggs ^ School Auchenshoogle High ^ Class 5W2)
```

This fact says that 'Joe Bloggs is a student at Auchenshoogle High in class 5W2'. All facts and rules in OPS5 are in brackets. The information inside the brackets is similar to the data in a record in a database. The field names are indicated by the caret mark ^. The database equivalent would be a record in the student table, like this:

Field names:	Name	School	Class
Field values:	Joe Bloggs	Auchenshoogle High	5W2

Here are two other facts in OPS5 about Joe Bloggs:

```
(Result ^ Name Joe Bloggs ^ Subject InfoSystems ^ Level Int2 ^ Grade B)
```

```
(Study ^ Name Joe Bloggs ^ Subject English ^ Level Higher)
```

These facts state that Joe Bloggs got a B result in Intermediate 2 Information Systems and that he is studying Higher English.

Here is a production rule in OPS5:

```
(p take-Higher-IS
  (Result ^ Name Joe Bloggs ^ Subject InfoSystems ^ Level Int2 ^ Grade B)
  →
  (MAKE Study ^ Name Joe Bloggs ^ Subject InfoSystems ^ Level Higher)
)
```

This rule says that *if* Joe Bloggs has a result of grade B in Intermediate 2 Information Systems, *then* a new fact is created that indicates that Joe Bloggs is studying Higher Information Systems.

If this rule is fired with the facts in working memory shown above, the condition of the rule (on the left-hand side of the implies operator \rightarrow) will match the working memory. As a result, the following new fact will be added to the working memory:

(Study ^ Name Joe Bloggs ^ Subject InfoSystems ^ Level Higher)

The following OPS5 rule will work for any student by making <N> stand for the name.

```
(p take-Higher-IS
  (or (Result ^ Name <N> ^ Subject ComputingStudies ^ Level SG ^ Grade 2)
      (Result ^ Name <N> ^ Subject InfoSystems ^ Level Int2 ^ Grade C)
  )
  →
  (MAKE Study ^ Name <N> ^ Subject InfoSystems ^ LevelHigher)
)
```

This rule says that to take Higher Information Systems, a student must have a result of either a grade 2 (or better) in Standard Grade Computing Studies *or* a grade C (or better) in Intermediate 2 Information Systems. If either of these facts is true, then a new fact is created that indicates that the student is studying Higher Information Systems.

OPS5 is important in the study of expert systems because it is one of the first knowledge representation languages which allowed knowledge to be represented in a simple way, and draw conclusions from that knowledge, without having to learn to program the computer. It also introduced many of the important features of forward-chaining systems: the working memory, pattern matching using the RETE algorithm and the conflict resolution strategies recency, specificity and refractoriness.

STRIPS

STRIPS stands for the Stanford Research Institute Problem Solver. The STRIPS program was developed in 1971 to allow a robot to plan a series of steps to move a number of objects around within a set of interconnected rooms. A robot called Shakey (Figure 10.1) was the first robot to be developed which could react 'intelligently' to its surroundings.

Figure 10.1: Shakey the robot



STRIPS was important because of the method it used to represent the knowledge about the locations of objects. Its method is still used in planning systems.

The state of objects in STRIPS was represented by a set of facts indicating the position and state of objects, for example:

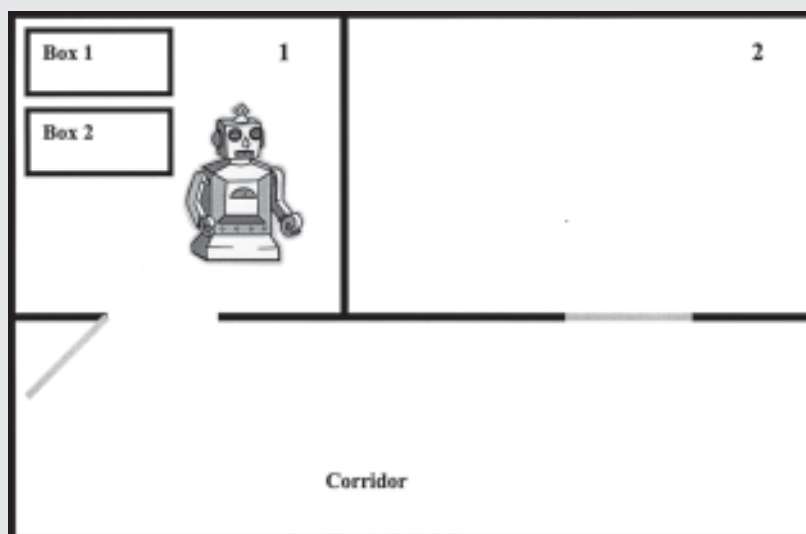
at(robot, room1) meaning the robot is in room 1

at(box1, room1) meaning box 1 is in room 1

at(box2, room1) meaning box 2 is in room 1

Figure 10.2 shows a diagram representing Shakey's world.

Figure 10.2: A map of Shakey's world



To carry out an action would result in a change to the set of 'at' statements. For example, suppose the robot were required to push box 1 from room 1 to room 2. This would be represented by the following rule:

push(box1, room1, room2):

pre-conditions: **at(robot, room1), at(box1, room1)**
 delete list: **at(robot, room1), at(box1, room1)**
 add list: **at(robot, room2), at(box1, room2)**

In everyday language, this rule means:

- To push box 1 from room 1 to room 2:
- check that the robot is in room 1 and box 1 is in room 1
- delete these facts from the working memory
- add two new facts to indicate that the robot and box 1 are in room 2

After this action had been carried out, the 'at' list would look like this:

at(robot, room2)
 at(box1, room2)
 at(box2, room1)

Here is a general push rule for moving any object (for example, X) from any room (Y) to any other room (Z):

push(X, Y, Z):

pre-conditions: **at(robot, Y), at(X, Y)**
 delete list: **at(robot, Y), at(X, Y)**
 add list: **at(robot, Z), at(X, Z)**

Actions in the STRIPS planning system were simply a case of working out the pre-conditions and which statements would be added to and deleted from the 'at' list.

Although the STRIPS system was only ever a research system, it is important in the study of expert systems because the way it represents actions is used in many planning systems for different domains.

Exercise 10.1 (Higher)

1. For each of the following expert systems, identify the *domain*, *category* and *purpose*:
 - (a) DENDRAL
 - (b) STRIPS
 - (c) MYCIN
 - (d) PROSPECTOR
 - (e) ONCOCIN

2. Describe the main contribution the following have made to the field of expert systems:
 - (a) MYCIN
 - (b) ONCOCIN
 - (c) OPS5

Chapter 11: Implications of using expert systems

This chapter looks at four aspects of using expert systems:

- the advantages and benefits
- the disadvantages and limitations
- the social, ethical and legal implications
- sources of error in expert systems

Advantages and benefits

The particular benefits gained from using expert systems are specific to the domain and purpose of each expert system. However, there are a number of general benefits that apply to all expert systems, including:

- dissemination of expertise: expert knowledge is readily available when needed
- productivity gains: expert systems improve productivity
- preservation of expertise: expert systems can be used to preserve valuable knowledge and experience
- training: expert systems can provide valuable learning experiences
- combining expertise of multiple experts: expert systems can contain the knowledge and experience of several human experts.

Dissemination of expertise

A human expert can only be in one place at any one time. His or her expertise is therefore only available to benefit a relatively small number of people. However, the expert knowledge contained within an expert system is readily available whenever it is needed.

For example, in remote areas of the Australian outback, medical expert systems are used by local nursing staff. The expert systems help to determine when it is necessary to call in the flying doctor service. They can help diagnose illnesses outside the nurse's normal expertise.

Expert systems used to assist in dealing with emergencies in a nuclear reactor are always available whereas human experts may not be – they may be at home, on holiday or even ill.

It is reassuring in such circumstances to have expert knowledge readily available.

Productivity gains

Expert systems improve productivity. For example, expert systems that are used to plan production schedules will identify weaknesses in the present system and suggest improvements that can be made.

Companies that use diagnostic expert systems have found that downtime due to faulty equipment is vastly reduced. They also find that troubleshooting costs are reduced and fewer experienced personnel are needed to repair sophisticated systems.

Preservation of expertise

Many companies have key personnel who are vital to the smooth running of the business. These key personnel are often major decision makers in the company. The knowledge they have about the business is important and difficult to replace – such expertise is a very marketable skill and many in similar positions are head hunted by rival companies. For these reasons, some companies build expert systems to hold the knowledge of these specialists and help in the decision making in their absence.

The knowledge contained in an expert system is permanent. Unlike human experts, who may retire, leave or die, the expert system's knowledge will last indefinitely.

Training

Medical expert systems in a doctor's surgery or in a hospital can assist in making a diagnosis. For inexperienced and recently qualified doctors, the confirmation provided by such an expert system is invaluable. In some cases, the doctor may come across an illness for the first time. The use of the justification features provides the doctor with an explanation of the diagnosis and gives reasons for asking questions. This, together with the use of certainty factors to express the system's belief in the diagnosis, combines to offer valuable learning experiences for an inexperienced doctor.

Combining the expertise of multiple experts

In many situations, the knowledge contained in an expert system has been gathered from more than one human domain expert. The knowledge base, therefore, contains the collective knowledge of several people – far more expert knowledge than one single domain expert.

This is the case, for example, in specialist areas of medicine. Many GPs have never treated a case of the meningitis virus – despite its publicity – and some fail to make an early diagnosis. The use of an expert system designed to assist in making an early diagnosis would be invaluable. Such an expert system would contain specialist knowledge from a number of experts in this field of medicine.

Furthermore, it is possible to combine in a single expert system the expert knowledge of a number of human experts in different domains. This would allow the expert system to draw conclusions that might otherwise be very difficult, if not impossible, to obtain.

Disadvantages and limitations

There are a number of disadvantages and limitations in using expert systems. These include:

- high development and maintenance costs
- a restricted domain
- a lack of common sense knowledge

High development and maintenance costs

Expert systems are very costly to develop. One reason for this is the amount of time it takes to create an expert system: gathering the specialist knowledge is a lengthy process and a domain expert's time is very valuable and therefore costly to obtain. Similarly, knowledge engineers are highly experienced and expert themselves, and therefore are highly paid. Testing and validating the final product is also very time-consuming.

In addition to these factors, there is the added problem of domain specialisation – a medical expert system that diagnoses skin infections has a very limited market. Similarly, an oil company that develops an expert system to help predict where to strike oil is unlikely to want their rivals to have the same expert system.

In other words, each expert system – by its very nature – is designed to solve problems in a specific area. This means that they are not as versatile as, for example, a word processing package, and they therefore have a very restricted market.

Developers of expert systems must cover the cost of their time – if there is only one likely customer, then that customer must cover the full development cost on their own.

Furthermore, in many specialist domains, knowledge is dynamic and frequently changing. For example, in the field of medicine, new drugs are constantly being developed and new cures are always being discovered. The knowledge in a medical expert system must be updated periodically to keep the expert system current. A human doctor would be able to learn these new facts and later use them to aid in diagnosis. An expert system, however, does not have this facility and all updates must be added by humans.

(Note that some expert systems have a capability to learn from experience and adapt their knowledge on the basis of previous consultations. However, the addition of new external facts must be carried out by humans.)

Restricted domain

All expert systems are designed to solve problems in a restricted domain. This means that although an expert system may contain more specialist knowledge than a single human expert, it has absolutely no knowledge of anything beyond that domain. For example, an expert system that is used to diagnosis faults in a car engine will be unable to provide even limited advice on what TV programme to watch. Human experts in the field of car maintenance will not only be able to diagnose the fault in the engine, they will also be able to suggest suitable a suitable TV programme to watch.

Lack of common-sense knowledge

Even though many expert systems contain the specialist knowledge of several human experts, they are unable to make decisions that require common sense. For humans, many everyday actions are so obvious that we tend not to think of them as intelligent. We make sensible decisions all the time without thinking about them.

For example, imagine you are walking along the street and you go to cross the road. You stop at the kerbside and check for moving

traffic. When it is clear, you cross the road. If you were asked why you stopped at the kerbside, the answer is so obvious that the question seems stupid.

An expert system, however, must be provided with all of the knowledge it contains. It is not possible for an expert system to have intuitive knowledge.

Social, ethical and legal implications

Responsibility for bad advice

Expert systems have been designed to replace human expertise, make decisions and give advice that would normally be suggested by a human expert. What happens when the advice is wrong? Or when the advice turns out to be bad advice? Who is responsible for this?

Consider the situation below.

A nuclear power station is generating electricity when suddenly a fault develops in the reactor. It is 3 o'clock in the morning. Only one specialist engineer is on duty – the others are home sleeping. Warning bells ring. Should the engineer close down the reactor? Should he call in the other specialists?

Recently, the company installed an expert system to help make these decisions. The expert system asks the engineer to provide information about the system. After considering the information provided by the engineer, the expert system suggests that it is a false alarm: one of the sensors is faulty and the readings provided by the sensor are incorrect. There is no need to close down the reactor – simply replace the faulty sensor.

The engineer follows this advice and the warning bells stop ringing. The expert system had correctly diagnosed the fault in the system.

But it could have worked out differently. Consider the following alternative ending to the story.

The engineer follows this advice and replaces the faulty sensor. The warning bells continue to ring. It appears that the sensor isn't faulty after all... the warning bells are still ringing... a full-scale emergency is brewing...

Next day, the headlines read:

‘EXPERT SYSTEM IS NO EXPERT’

‘NUCLEAR FALLOUT DESPITE ADVICE FROM THE EXPERT’

Who is to blame for this disaster?

Who must take the responsibility for this bad advice? The options are:

- the team of software engineers who developed the expert system
- the engineer who relied on the advice suggested by the expert system
- the human experts who supplied the specialist knowledge contained in the expert system
- the power company which installed an expert system to replace human expertise.

Let’s consider each of the suspects in turn:

- *The team of knowledge engineers who developed the expert system*

The team who developed the expert system were responsible for representing the specialist knowledge in the expert system as facts and rules. They were also responsible for carrying out tests to make sure that the system was operating correctly. There is scope for error.

They may have represented the specialist knowledge incorrectly.

They may not have tested the system adequately.

- *The engineer who relied on the advice suggested by the expert system*

The engineer is a specialist in this area. He or she was responsible for providing the expert system with readings and information about the status of the reactor. It is likely that he or she was under immense pressure. There is, once again, scope for error.

He or she may have responded incorrectly to questions asked by the expert system.

He or she should have had the professional integrity to call for a second opinion and not blindly act on the advice from an expert system.

- *The human experts who supplied the specialist knowledge contained in the expert system*

The human experts who supplied the knowledge engineers with the specialist knowledge are responsible for providing knowledge that is accurate. They must also take part in the testing of the system to make sure that the advice suggested by the expert system matches that advice that they would give in the same circumstances. Again, there is scope for error.

They may have provided inaccurate information.

They may have provided incorrect advice during testing so that errors were not discovered.

- *The power company which installed an expert system to replace human expertise*

The power company are responsible for the safety of the reactor at all times and they decided to install an expert system to replace scarce human expertise. The company bought the services of a specialist knowledge engineering team and provided them with a detailed specification for the expert system. They provided the human experts and allowed time for the knowledge engineering team to meet with them. Scope for error includes:

- not enough liaison time was made available
- emergency procedures and safety guards were not updated when the expert system was installed
- the system wasn't properly phased in to accommodate testing.

It is clear that it is possible to lay the blame with a variety of groups or individuals and each of these can equally well provide good reasons why they are not to blame and the others are.

Users of expert systems must always remember that they can, if they wish, ignore the advice of the expert system when making a final decision. Expert systems are only electronic consultants – it is the user who has control over the final solution and its application.

Sources of error

When developing expert systems, there are a number of possible errors that may occur and could result in the expert system failing to operate correctly (crashing) or giving incorrect advice.

The domain expert

It could be that the domain expert has made an error in the knowledge supplied to the knowledge engineer. A side-effect of the development process is the potential detection of incorrect knowledge.

For mission-critical projects, in which human life or property is at stake, it is necessary to validate the expert's knowledge, usually by involving one or more other domain experts, if available, in reviewing the knowledge obtained. Of course, this is an expensive option, but one in which the cost is justified.

Mis-interpretation of expert knowledge

Because the knowledge engineer is unlikely to be familiar with the domain of knowledge of the human expert, it is possible that the domain expert's knowledge may be misunderstood or misinterpreted.

For example, suppose a fire safety expert says 'You can extinguish a fire with water', and the knowledge engineer interprets this as 'All fires can be extinguished with water'.

These semantic errors can occur if the knowledge engineer misinterprets the expert's answers, or if the expert misinterprets the knowledge engineer's question (or both).

Programming

Errors can occur at the programming stage if a rule is incorrectly programmed into the knowledge base by the programmer. This can occur if the knowledge provided to the programmer by the knowledge engineer is unclear or is insufficiently structured. It can also occur as a result of errors earlier in the knowledge acquisition process.

Simple syntax errors can occur if a rule is entered incorrectly in the knowledge base by the programmer. However, the expert system shell should detect and report these errors, as shown in Figures 11.1 and 11.2.

Figure 11.1: A syntax error in the InterModeller expert system shell

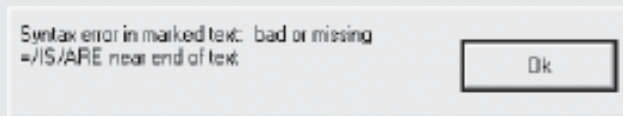
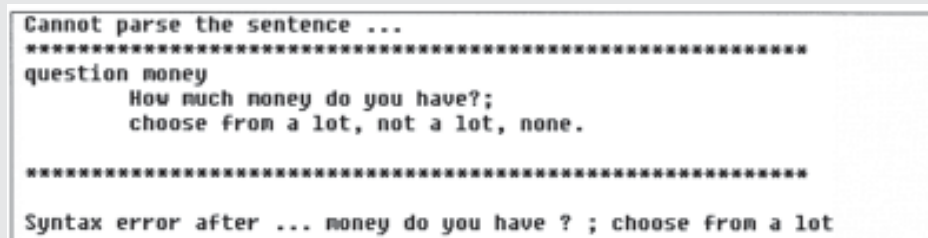


Figure 11.2: A syntax error in the flex expert system shell



Inferencing

There may be errors that result from the chain of inference being followed. For example, there may be unexpected interactions between rules, particularly if certainty factors are involved.

In addition, as with all software, it is possible that the inference engine in the expert system shell may have bugs. As with any complex software, it is virtually impossible to guarantee that the software is 100% bug free.

Furthermore, such bugs may be uncommon and difficult to detect. For example, there may be a bug that shows up only if there are 37 rules in the conflict set. Bugs are particularly difficult to detect and correct if they are inconsistent.

Limits of ignorance

A problem that is common to all stages in the development process is specifying the *limits of ignorance* of the system. Human experts (generally) know the extent of their knowledge and will admit when they are uncertain or don't know the answer. However, unless an expert system is programmed to admit uncertainty, it may provide answers even when there is insufficiently strong reason to do so.

Exercise 11.1 (H)

1. Expert systems are good at predicting results. PROSPECTOR is an expert system that interprets geological data and is used to help locate ore deposits and minerals. It contains rules about ore environment and conditions as well as data on rocks and minerals.
 - (a) What advantages are there for a mining company using PROSPECTOR?
 - (b) What advantages do you think PROSPECTOR might have for the environment?

2. In conventional aircraft, computers are used for navigational purposes and to control fuel flow. The greater part of long-distance flights is handled by autopilots relying on computer systems. The pilot, however, can over-ride the computer. The Airbus series of aircraft was designed to incorporate fly-by-wire technology. In such systems, computers are provided with all available human expertise in flying an aircraft. The system then controls the aircraft engines; there are no manual connections or mechanical back-ups. The computer system may over-ride the pilot.
 - (a) What advantages are there from the safety point of view of having the pilot's decisions over-ruled by the computer?
 - (b) What disadvantages are there?
 - (c) Whom would you blame if there was an accident:
 - the pilot
 - the hardware supplier
 - the software firm
 - the airline?Give reasons for your answer.
 - (d) What financial gains might be made by airlines using fly-by-wire technology? Why?
 - (e) What difficulties would face teams designing expert systems for new fly-by-wire aircrafts?

3. Some supermarkets are using expert systems to monitor customers' spending patterns. Each customer has a points card that is swiped at the checkout and their purchases are recorded. This data is then analysed by expert systems and trends are

identified. Customers can then be targeted with special offers and money-off vouchers for products that they are likely to buy.

- (a) Do you consider that this is justified or is it an invasion of privacy? Explain your answer.
 - (b) Some of these companies sell the information gathered to prospective advertisers. Discuss the benefits and drawbacks of doing this.
4. You've been feeling unwell and decide to make an appointment with the doctor. You arrive at the surgery in good time and sit in the waiting room. At last it is your turn. When you walk into the doctor's consulting room, you notice the computer on the desk. The computer is running a medical expert system. During your visit, the doctor asks you questions suggested by the expert system and enters your responses. Eventually, the expert system provides a diagnosis and the doctor writes out a prescription.
- (a) What advantages are there in the system described above? What disadvantages are there?
 - (b) Do you think a newly qualified doctor's attitude to the expert system would be different to the attitude of a more experienced doctor?
 - (c) If an expert system gives a wrong diagnosis, who is morally responsible? Give reasons for your answer.
 - (d) Do you think a computer could ever replace a doctor?
 - (e) A medical expert system could be very helpful in developing countries. Explain why.

Glossary

Intermediate 2 level

The following are technical terms that candidates at Intermediate 2 and Higher levels are expected to be familiar with and able to use and interpret correctly in context. (Words in bold type are also defined in this glossary.)

Advice

One category of **expert system**. Also used more generally to refer to the **conclusion** of a **consultation**. See *Type of expert system*.

Backward chaining

A form of **inferencing** in which the system selects a **goal** or **hypothesis** and then attempts to find evidence (**facts**) to support it.

Chaining

The method of **inferencing** used by the expert system's **inference engine**. See *Backward chaining, Forward chaining*

Classification

One category of **expert system**. See *Type of expert system*.

Conclusion

The final **advice** or result of a **consultation** with the **expert system**

Consultation

The process of seeking **advice** from an **expert system** by answering **questions** posed by the system

Data driven

Describes a **forward-chaining** system in which the known **facts** are used to determine the **rules** that are applied in order to reach a **conclusion**.

Diagnosis

One category of **expert system**. See *Type of expert system*.

Domain

The field of **knowledge** or expertise of the **expert system**

Domain expert

A human expert with **knowledge** or expertise in a particular **domain**

Expert system

A computer program that contains the specialist **knowledge** of one or more human experts in a form that others may use to solve problems in a specific **domain**. It provides the **user** with **advice** via a **consultation** and can explain the **advice** it gives and why it is asking particular **questions**.

Expert system shell

A piece of software with a built-in **inference engine** and **user interface**, but with an empty **knowledge base** (i.e. containing no **facts** or **rules**).

Explanation

A function of an **expert system** that makes clear the reasoning it has followed. A 'How' explanation provides reasons for the conclusion reached. A 'Why' explanation provides reasons for the current question being asked.

Fact

A statement contained in the **knowledge base**. May be pre-entered or provided by the **user** during a **consultation**.

Factor table

A method of structuring **knowledge** in the form of a table. Each row of the table corresponds to a conclusion, and each column corresponds to an attribute, condition, or question.

Forward chaining

A form of **inferencing** in which the system uses the known **facts** to determine the **rules** to be applied in order to reach a **conclusion**. See *Data-driven, Working memory*.

Goal, goal-driven

The **conclusion** that the system is attempting to reach in a **backward-chaining** system. Such a system is described as goal driven because it starts by selecting a goal or **conclusion** and then attempts to find evidence to support it.

How

See *Explanation*.

Hypothesis

The **conclusion** or **goal** that the system is attempting to reach in a **backward-chaining** system.

Inference engine, inferencing

The component of the **expert system** that determines the order in which the **rules** in the **knowledge base** are applied, and therefore also the order in which **questions** are asked of the **user**.

Justification

See *Explanation*.

Knowledge

The ability to use known **facts** to draw **conclusions** and make decisions. Created by a logical process in which new **facts** are derived from known **facts** by the application of **inference** rules.

Knowledge acquisition/elicitation

The process of obtaining expert knowledge from a **domain expert** or other source. See *Knowledge engineer*.

Knowledge base

Contains the **facts** and **rules** that represent the specialist **knowledge** contained in the **expert system**. This **knowledge** is written in an appropriate **knowledge representation language**.

Knowledge engineer

The person responsible for obtaining expert knowledge from a **domain expert** or other source and structuring the knowledge in a way that can be used within an **expert system**. See *Knowledge acquisition, Knowledge representation, Programmer*.

Knowledge representation

The method used to organise and structure a **domain expert's knowledge**. See *Factor table, Production rule*

Knowledge representation language (KRL)

The method used by an **expert system shell** to input **facts** and **rules** into the **knowledge base**. Each expert system shell has its own KRL.

Planning

One category of **expert system**. See *Type of expert system*.

Production rule

A form of **knowledge representation** commonly used in **expert systems**. Rules typically have the form IF <conditions> THEN <actions>.

Programmer

The person responsible for entering **facts** and **rules** into the **knowledge base** of an **expert system shell** based on information provided by a **knowledge engineer**.

Query

The process of extracting **knowledge** from the **knowledge base**.

Question

The method used by most **expert systems** to obtain information from the **user** during a **consultation**.

Rule base

See *Knowledge base*.

Type of expert system

The categories of **expert system**: advice, classification, diagnosis and planning.

User

A person obtaining **advice** from an **expert system** by **consultation**.

User interface

The component of an expert system that is responsible for asking the user **questions**, obtaining answers, displaying **conclusions** and explaining its reasoning.

Validation

A method of ensuring that the **conclusions** derived by an **expert system** correspond correctly to those given by a human expert.

Why

See *Explanation*.

Working memory

Contains the set of known **facts** in the **knowledge base** (usually describes a **forward-chaining** system).

Higher

The following are technical terms that candidates at Higher level are expected to be familiar with and able to use and interpret correctly in context.

Certainty factor

A number, representing a percentage value, representing the degree of **uncertainty** in a rule or fact. May be a whole number between 0 and 100, or a decimal number between 0 and 1.

Conflict resolution

During forward chaining, the process of selecting a rule from a **conflict set**.

Conflict set

In a forward-chaining system, the set of rules whose conditions currently match the facts in **working memory**.

Context limiting

A **conflict resolution** strategy in which sets of rules are grouped together (called 'agendas') and may be switched at different stages of the inferencing process (called a 'context'). Also known as *setting a rule agenda*.

Data ordering

A **conflict resolution** strategy in which each fact in **working memory** is given a value to represent its importance or priority. Rules that use the most important facts are considered in preference to others.

Decision tree

A form of knowledge representation consisting of a number of choice points representing decisions to be made (or questions to be answered), with branches representing the available choices.

Deductive database

Software that combines the storage and retrieval capability of a database with the inferencing capability of an expert system.

Existential quantifier

The \exists symbol used in predicate logic, meaning 'there exists'. See *First order logic*.

Expertise

The knowledge and experience of a human expert, which an expert system is designed to emulate.

Fire

In a forward-chaining system, when a rule is selected from a **conflict set** and is carried out it is said to 'fire' or 'be fired'.

First-order (predicate) logic

A form of knowledge representation in which statements are represented as predicates, with variables that can be used to stand for different values.

Heuristic

A rule of thumb that is used to guide an expert system towards a solution.

Implication operator

The \rightarrow symbol used in **logic**, meaning 'implies'.

Limits of ignorance

Knowing the extent of your knowledge and being able to admit when a conclusion is unknown or uncertain. Generally, a characteristic of a human expert that is difficult for expert systems to emulate.

Logic

A type of knowledge representation. See *First-order (predicate) logic*, *Zero-order (propositional) logic*.

Predicate

A statement of fact used in **first-order (predicate) logic**.

Proposition

A statement of fact used in **zero-order (propositional) logic**.

Recency

A **conflict resolution** strategy which considers those rules which use the facts most recently added to the **working memory** in preference to others.

Refractoriness

A **conflict resolution** strategy that avoids looping by preventing rules from repeatedly **firing**.

RETE algorithm

An efficient algorithm for determining a **conflict set** by maintaining a list of matches between facts and conditions.

Rule ordering

A **conflict resolution** strategy in that rules are applied in the order in which they are listed in the knowledge base. Also known as first-come-first-served.

Specificity

A **conflict resolution** strategy which considers the most specific rules, i.e. those with the largest number of conditions, in preference to others.

Uncertainty

The degree of confidence, or lack of it, in a fact, rule or conclusion. Expressed using a **certainty factor**.

Universal quantifier

The \forall symbol used in predicate logic, meaning 'for all'. See *First-order logic*.

Zero-order (propositional) logic

A form of knowledge representation in which statements are represented as propositions, with implications between them.

Index

- 3GL 61
 4GL 61
 algorithm 17
 artificial intelligence 10
 attribute–value pairs 29
 backward chaining 33
 INTERNIST 107
 certainty factor 100
 MYCIN 104
 conflict resolution 96
 conflict resolution strategy
 context limiting 98
 data ordering 98
 first-come-first-served
 See rule ordering
 OPS5 109
 recency 97
 refractoriness 97
 rule agenda
 See context limiting
 rule ordering 97
 size ordering
 See specificity
 specificity 97
 conflict set 96
 consultation 12
 data driven 40
 INTERNIST 107
 R1/XCON 106
 data ordering 98
 databases
 and expert systems 16
 deductive 17
 decision tree 60, 81
 DENDRAL 105
 domain expert 10, 54, 63
 domain of expertise 10
 expert 9
 expert system 9, 12, 18, 104
 and databases 16
 development cycle 55
 shell 15, 61
 explanation 50
 factor table 28, 60
 first-order logic
 See logic, predicate
 forward chaining 33, 39, 96
 INTERNIST 107
 OPS5 109
 R1/XCON 106
 goal driven 34
 heuristic 17
 hypothesis 33, 51
 inference engine 13, 33
 INTERNIST 107
 knowledge 15
 knowledge acquisition 56, 57
 ONCOCIN 105
 knowledge elicitation 57
 knowledge engineer 54, 60
 knowledge
 representation 28, 55, 60
 knowledge representation
 language 27, 60
 KRL
 See knowledge representation
 language
 limits of ignorance 122
 LISP 61
 logic
 predicate 90
 propositional 85

MYCIN	104	RETE algorithm	96, 98
ONCOCIN	105	OPS5	109
OPS5	109	rule agenda	
predicate logic		<i>See</i> rule ordering	97
<i>See</i> logic, predicate		rule tree	50
production rule	26, 60	size ordering	
production system		<i>See</i> specificity	
OPS5	109	specificity	97
programmer	54	OPS5	109
programming language		STRIPS	110
fourth-generation		system validation	55, 63
<i>See</i> 4GL		uncertainty	99, 104
third-generation		user	54
<i>See</i> 3GL		working memory	39, 96
Prolog	61	XCON	
propositional logic		<i>See</i> R1	
<i>See</i> logic, propositional		zero-order logic	
PROSPECTOR	108	<i>See</i> logic, propositional	
R1	106		
recency	97		
OPS5	109		
refractoriness	97		
OPS5	109		



SECTION 4**Chapter 1****Exercise 1.1**

1.
 - (a) Cookery
 - (b) Football
 - (c) Motor cars
 - (d) Natural history
 - (e) Films
 - (f) Tennis
 - (g) Astronomy
 - (h) Astrophysics
 - (i) Information Systems (hopefully!)

2. There are several correct answers to each question. Those given here are examples only.
 - (a) Alan Titchmarsh, Charlie Dimmock
 - (b) Lawrence Llewellyn Bowen
 - (c) Michael Fish, John Kettle
 - (d) Radio DJs

Exercise 1.2

1. An expert system is a computer program that:
 - contains the specialist knowledge of one or more human experts; this expert knowledge is in a form that others may use to solve problems in a specific domain
 - provides the user with advice
 - can explain the advice it gives and why it is asking particular questions.

2. Human experts.

3. Any two parliamentary experts, for example, your local MP, the Speaker of the House of Commons, the Prime Minister, etc.

4. Knowledge base, inference engine, user interface.

5. Advice and justification.

6. (a) Advice
(b) Classification
(c) Planning
(d) Diagnosis
(e) Advice
(f) Advice
(g) Advice/diagnosis
(h) Classification
(i) Planning
7. (a) Advice
(b) Classification
(c) Planning
(d) Advice/diagnosis
(e) Diagnosis
(f) Classification
8. (a) The process of authorising card transactions was very time-consuming and people-intensive.
(b) There were a large number of rules.
(c) So that they could continue to use the system without having to be retrained in its use.
(d) Saving in employment costs, reduction in fraudulent use.
9. (a) There are too many problems that can occur in different product lines for a human to remember.
(b) (i) Reduces the amount of time on the phone, so lower phone bills
(ii) More problems are solved in the time available, therefore there is an improvement in customer satisfaction.

Exercise 1.3

1. (a) To help its auditors keep up to date with changes in tax regulations.
(b) By comparing the advice of the expert system with the advice of the internal experts.
(c) The tax regulations are complex and detailed, so each expert had his/her own field of expertise.
(d) The hours spent gathering expert knowledge from the company's experts.
(e) They had access to a text expert in the office whenever it was needed.

- (f) The expertise was more widely available, so saving the amount of time spent by the human experts (e.g. in travelling between offices).
2. (a) Engine repairs were causing wastage of components.
 (b) Customers got their cars back sooner.
 (c) A more precise repair could be carried out, thus reducing wastage and cost. This led to greater levels of customer satisfaction.
 (d) A natural language interface was developed to help mechanics, many of whom were discovered to be dyslexic.

Chapter 2

Exercise 2.1

1. Two possible answers, depending on the cause and effect:
- (i) IF food is healthy
 THEN food is full of vitamins
 AND food is full of protein.
- (ii) IF food is full of vitamins
 AND food is full of protein
 THEN food is healthy.
2. IF the season is winter
 AND the night sky is clear
 THEN there will usually be a frost in the morning.

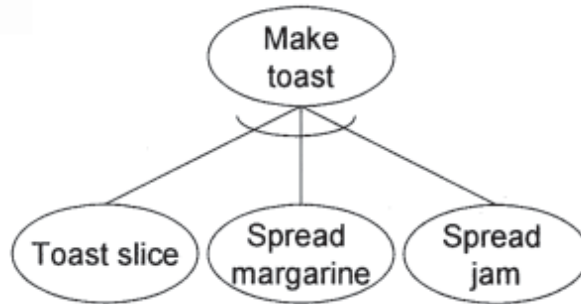
3.

Cloud type	Level	Appearance	Likelihood of rain
Cirrus	High	Wispy	Soon
Altostratus	Medium	Thick layer blocking out the sun	Unlikely
Nimbostratus	Low	Flat, featureless layer	Frequently

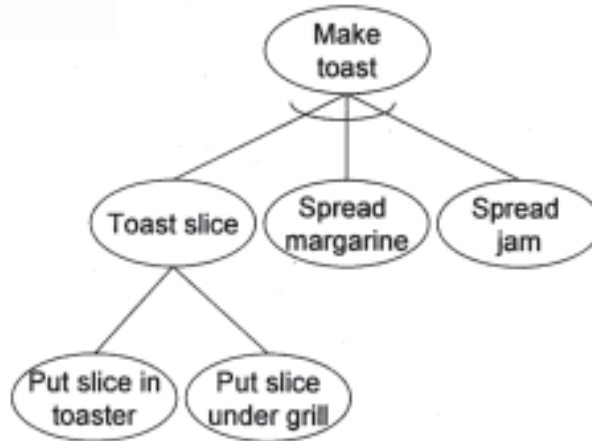
Chapter 3

Exercise 3.1

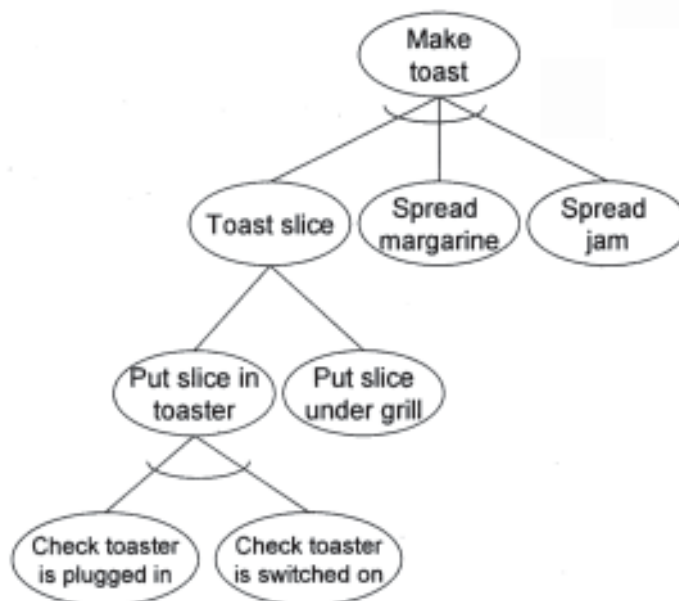
1. (a)



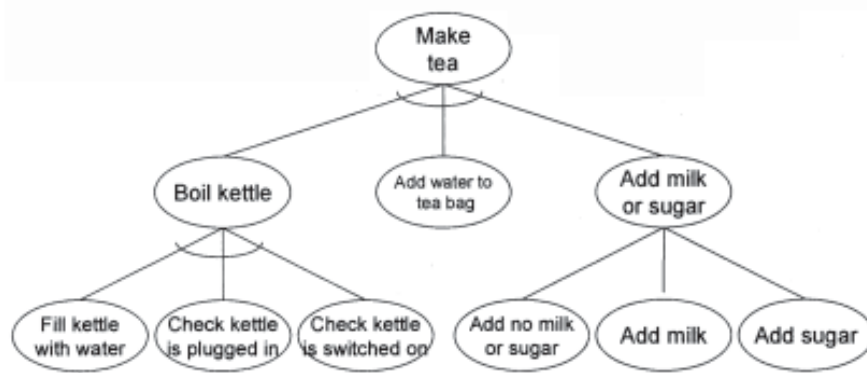
(b)



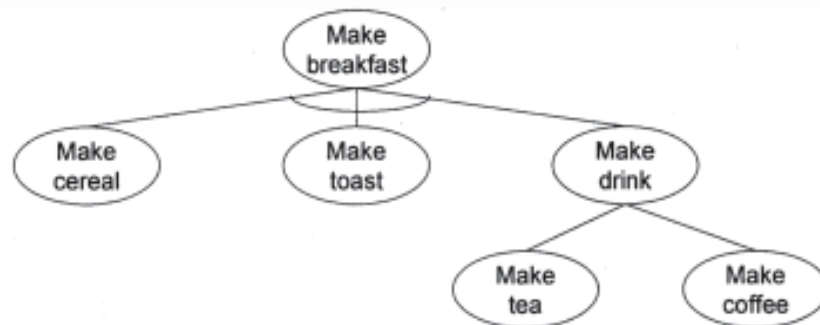
(c)



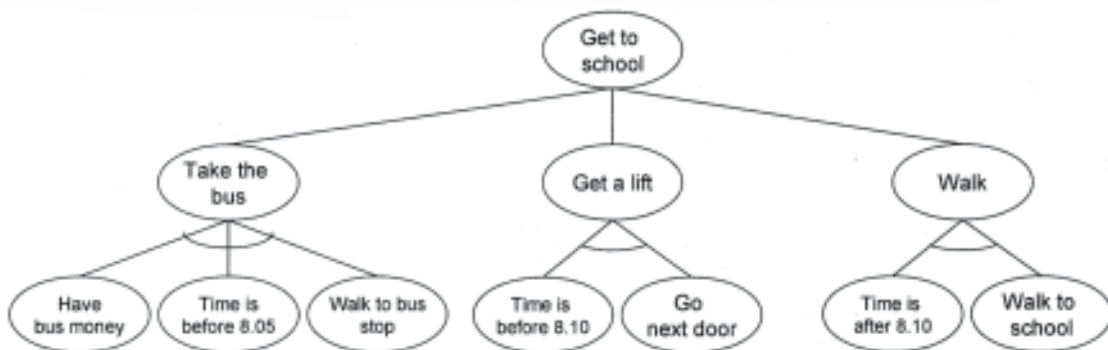
2. (a) There are a number of possible correct answers, depending on how much detail is given. The following is one possible answer.



- (b) The sub-goal 'make tea' would need to change to 'make drink' with two alternatives, 'make tea' and 'make coffee', as shown below.



3. The following is one valid solution. There are alternative correct solutions.



Exercise 3.2

1. The investigation begins with the evidence from the crime scene and works forward from there to draw conclusions (forward chaining). However, if the initial investigation is unsuccessful, and a conclusion cannot be drawn from the evidence, detectives may have to form alternative hypotheses about the suspect, motive, etc. and then search for further evidence to support the hypotheses (backward chaining).

Chapter 4**Exercise 4.1**

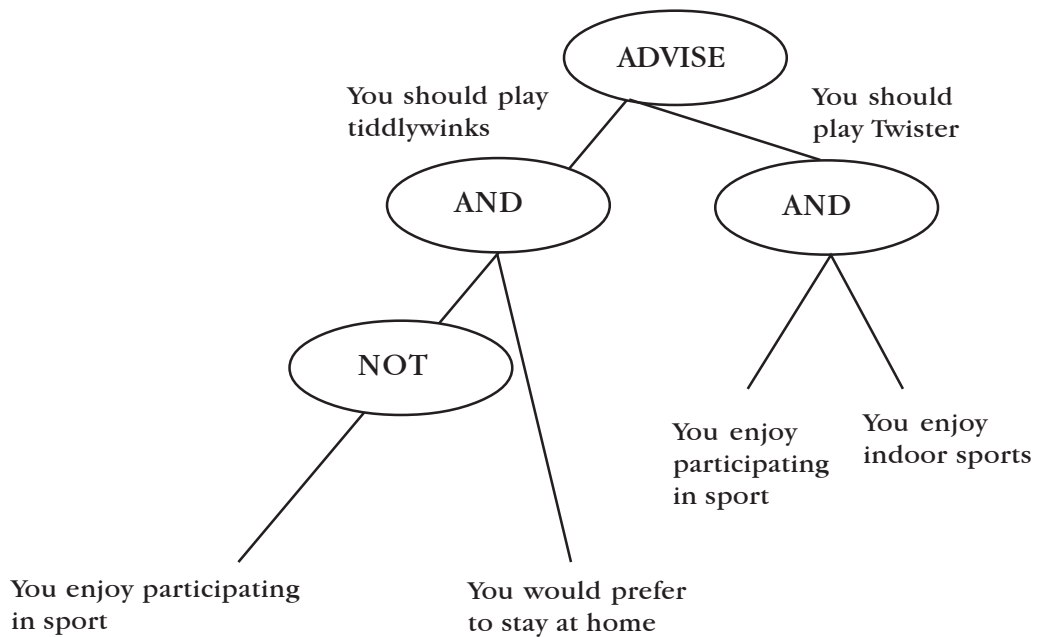
1. Text, table of results, graph.
2. Descriptions of 'How' and 'Why' explanations.

Exercise 4.2

1. The functionality of the expert system depends on its ease of use. Its ease of use is determined by the user interface. The software development must take the time to make sure that the user interface is simple and easy to use – otherwise the user will be unproductive.
2.
 - (a) The 'How' explanation facility can show that the reasoning used for the conclusion reached matches that of the domain expert.
 - (b) If the conclusion reached is incorrect, the 'How' explanation facility can show how the expert system's reasoning was applied, and how it differs from that of the domain expert. In turn, this can enable the designer to correct the error.
3.
 - (a) The 'How' explanation can reassure the patient and doctor that the conclusion reached is correct and is supported by the medical evidence. The 'Why' explanation can also reassure the patient that only relevant questions about the patient's symptoms or medical history are being taken into account.

- (b) The 'How' explanation can reassure the investor that the conclusion reached is correct and that the investment is likely to be secure.
4. (a) When the user wants to know how the expert system has reached a particular conclusion.
(b) When the user wants to know why the expert system is asking a particular question.
5. Without the ability to justify its actions, the advice from an expert system would not be taken seriously. If it is to replace human expertise when that expertise is scarce, then the user must be confident that the expert system is knowledgeable. In the same way that a human expert can explain his/her actions, an expert system must be able to do so likewise.
6. (a) 'You should play tiddlywinks'.
(b) Do you enjoy participating in sport?
(c) Because I may be able to advise that you should play tiddlywinks.
(d) The expert system will reject this goal because it can no longer confirm it. It will then search the rule tree for an alternative goal to test.
(e) The expert system would have tested the second condition by asking 'Do you want to stay at home?'.
7. (a) Any appropriate conditions are acceptable. For example:
Condition 1: You enjoy participating in sport.
Condition 2: You want to stay at home/stay indoors.

(b)



(c) Two

(d) You should play Twister.

(e) Because you enjoy participating in sport and you want to stay at home.

Chapter 5

Exercise 5.1

1. Knowledge acquisition, knowledge representation, system validation.
2. Domain expert, knowledge engineer, programmer, user.
3.
 - (a) User interface
 - (b) Knowledge base
 - (c) User interface
 - (d) User interface

Exercise 5.2

1. Knowledge acquisition: gathering of expert domain knowledge. Usually involves interviewing and observing one or more human experts.
Knowledge representation: selection of a suitable KRL and then producing the facts and rules that represent the domain knowledge.
System validation: testing the system to make sure that it is correct. Usually involves comparing advice produced with that provided by human experts.
2. Understanding the domain knowledge and dealing with uncertainty.
3. Traditional programming languages (3GLs); specialist AI languages (4GLs); expert system shells.
4. Inference engine is built in and doesn't have to be programmed and tested.
5. The team of knowledge engineers. The expert system produced by a shell will look and behave in the same way as other expert systems produced by the same shell. It will seem to the finance company that their product is just the same as another product. However, the knowledge contained in the knowledge base will be unique to that company. Only the inference engine and user interface will be the same.
6. By comparing the advice produced with the advice suggested by human domain experts. Only when the advice produced consistently matches the advice of human experts will the system be considered to be valid.
7. Interview experts on the dig. Represent the knowledge gained in the KRL used to develop the original expert system. Add the additional knowledge to the knowledge base. Test and validate the new rules and facts in the system.
8. By interviewing people who are expert in the field of European wild flowers. Follow-up research may be necessary in order to confirm facts and ensure a full understanding of the information provided.

Chapter 6

Exercise 6.1.1

A correct solution will depend on the expert system shell and implementation method employed. The solution should correctly identify each of the vegetables.

The following are sample rules which may be appropriate for implementation in an expert system shell.

```
IF colour is white
AND location_of_growth is underground
AND shape is long_and_pointed
THEN vegetable is parsnip.
```

```
IF colour is green
AND location_of_growth is above_ground
AND shape is round
THEN vegetable is cabbage.
```

```
IF colour is orange
AND location_of_growth is underground
AND shape is long_and_pointed
THEN vegetable is carrot.
```

Exercise 6.1.2

```
IF colour is white
AND location_of_growth is above_ground
AND shape is round
THEN vegetable is onion.
```

```
IF colour is green
AND location_of_growth is above_ground
AND shape is long
THEN vegetable is leek.
```

Exercise 6.2

A correct solution will depend on the expert system shell and implementation method employed. The solution should correctly advise each of the spare-time activities.

Exercise 6.3

A correct solution will depend on the expert system shell and implementation method employed. The solution should show evidence of knowledge acquisition, knowledge representation (as a factor table with refinements) and implementation, and correctly advise each of the spare-time activities.

Chapter 7**Exercise 7.1**

A correct evaluation should cover all the aspects mentioned in this chapter. The evaluation should be graded by the quality of response.

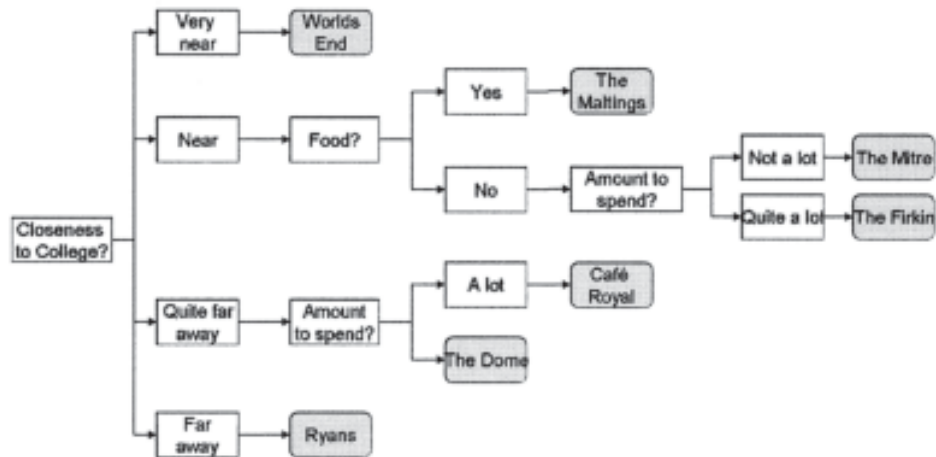
Exercise 7.2

A correct evaluation should cover all the aspects mentioned in this chapter, including the additional aspects required of a detailed evaluation. The evaluation should be graded by the quality of response.

Chapter 8

Exercise 8.1

- The following is a sample solution. There are other correct solutions depending on the order of selection of the attributes.



- The correct solution will depend on the factor table produced. However, the solution should be checked for the following:
 - a root node
 - conclusions on the leaves
 - nodes representing choice points (i.e. conditions in the factor table)
 - branches representing possible choices (i.e. values for conditions in the factor table)
 - all nodes and branches labelled.

Exercise 8.2

- $Q \wedge P$
 - $Q \wedge S \rightarrow T$
 - $\neg S \rightarrow \neg T$
 - $\neg Q \vee \neg S \rightarrow P$
- Yes.
 - Yes. (However, if there is no-one called Socrates, then the first sentence would be false, but the second sentence would be true.)

- (c) No. If you have one person who is telling the truth, and another one who isn't, then the first statement is true, but the second statement is false.
- (d) Yes.
- (e) No. If it is possible both to look and not look, then the first statement is false, but the second statement is true (i.e. it is false that you must not look (because you are allowed to look), but it is not true that you must look (because you are allowed to not look!).
- (f) Yes, if the first sentence is taken literally as read. However, the first sentence is usually understood to mean 'I think he isn't coming'. If so, the answer is 'No', because if I don't know whether he is coming or not, then the first sentence is false, but the second sentence is true!
3. (a) Yes.
- (b) No. The second sentence doesn't indicate that Mary and Jane are friends *of each other!*
- (c) Yes. The first sentence doesn't make clear who Jane lives next door to.
- (d) No. Suppose that John lives next door and that Mary went to visit someone in the next street. Then the second sentence would be true, but not the first!
- (e) Debatable. If Mary and John had three children and then got married, the second sentence would be true. Would the first be false? There is certainly some sort of implication that they were married first and then had three children!
- (f) Yes.
- (g) No. The problem is that whether something counts as small or not depends on what it is being compared with. In the first sentence it is clear that Dumbo is being compared with elephants. So that sentence will be true if he is small for an elephant. In the second sentence he may be being compared with animals in general. In that case the second sentence is presumably false (as even small elephants are big animals!).
- (h) Yes. The AND operator \wedge can usually be used to represent 'but'. Note that in this example the use of 'but' also suggests that Dumbo's intelligence is in contrast with the fact that most elephants are not quite intelligent.
- (i) No. To say that Dumbo is a skilful flier is to say that he is skilful at flying. To say that Dumbo is skilful (simply) may be to say that he is skilful at a wide range of things, or that he is skilful at some specific thing, which may not be flying. So, the first sentence would be true and the second false, if he were

skilful at flying, but not at a wide range of things (or not at the specific thing in question). Equally the second sentence would be true but the first false if, say, he were skilful at a wide range of things that did not include flying.

4. (a) Yes.
- (b) No. The first sentence will be false, but the second true, if he comes both days. (Does this mean that the 'or' is exclusive here? No! The presence of 'but not both' is enough to make the sentence as a whole exclusive. But the 'or' could perfectly well be inclusive. So the whole sentence could be translated as '[John will come today \vee John will come tomorrow] \wedge \neg [John will come today \wedge John will come tomorrow]').
- (c) Yes. It is correct to treat this as equivalent to an inclusive 'or'. To treat it as exclusive would be to take it as meaning the same as 'You won't win unless you try, in which case you will win.'
- (d) It depends on whether you believe that being the President of the United States makes you the most powerful man in the world. If the first sentence means the same as 'He is president of the United States; that is, the most powerful man in the world.' the translation will not do. For then, if he were the most powerful man in the world, but not president of the United States, the second sentence would be true, but the first false.
- (e) No (probably). If you may have soup but not fish, the first sentence (as most commonly used) will be false but the second true.
- (f) Yes.
5. (a) Yes.
- (b) No. The first sentence means that John doesn't know whether Liverpool won the cup. Even if we take 'John doesn't know' to mean 'John does not know that Liverpool won the cup', the second sentence does not capture what the first sentence says. It would, for instance, be true if Liverpool did not win the cup and John knows that they did not; but the first sentence would be false.
- (c) Yes. We could equally have had 'You will succeed \rightarrow you will try'.
- (d) No. The first sentence would be true if trying were necessary for your success but not sufficient to guarantee it, whereas the second sentence would be false. Equally if trying were sufficient for success but not necessary, the second sentence

would be true but the first false. An acceptable translation would be 'You will succeed \rightarrow you will try' or (equivalently), ' \neg You will try $\rightarrow \neg$ You will succeed'.

- (e) Yes. Or 'You will succeed \rightarrow you will try'.
- (f) No (most likely). The original sentence means (most probably) that I will be in the garden whether my mother telephones or not; that is, it will be false if I will not be in the garden. The second sentence, however, will be true if I will not be in the garden and my mother does not telephone.
- (g) Yes.
- (h) No. Most likely the first sentence is saying something general about eggs: any egg that has been boiled for 10 minutes is hard. In that case it will be false if some eggs boiled for 10 minutes are hard and some are not. The second sentence, however, will be true if there is an egg which has been boiled for 10 minutes and there is an egg which is hard (even a different one).

Exercise 8.3

1. (a) $\text{plays}(\text{Amy}, \text{violin}) \wedge \text{plays}(\text{Amy}, \text{recorder})$
 (b) $\text{plays}(\text{Sam}, \text{piano}) \rightarrow \text{plays}(\text{Sam}, \text{harpsichord})$
 (c) $\forall x: \text{plays}(x, \text{bagpipes}) \rightarrow \text{likes}(\text{susan}, x)$

2. (a) $\forall x: \text{human}(x) \rightarrow \text{man}(x)$
 (b) $\forall x: \text{woman}(x) \rightarrow \neg \text{likes}(x, \text{John})$
 (c) $\forall x: \text{likes}(x, \text{John}) \rightarrow \text{woman}(x)$
 OR $\forall x: \neg \text{woman}(x) \rightarrow \neg \text{likes}(x, \text{John})$

Note: The inverses of the above are incorrect because there could be some women who do not like John:

- $\forall x: \text{woman}(x) \rightarrow \text{likes}(x, \text{John})$
- $\forall x: \neg \text{likes}(x, \text{John}) \rightarrow \neg \text{woman}(x)$
- (d) $\forall x: \text{human}(x) \rightarrow \text{man}(x) \vee \text{woman}(x)$
- (e) $\forall x: \text{woman}(x) \rightarrow \text{likes}(x, \text{John})$

Chapter 9

Exercise 9.1

1. (a) 'may', 'is likely'
(b) 'perhaps', 'a kind of', 'managed'/'some'
2. There are many answers, including possible, probably, unlikely, doubtful, often, seldom.
3. The domain may be imprecise; the human expert may be not be 100% certain of his/her knowledge; the user may provide data to the expert system that is uncertain or incomplete.
4. (a) 2, 3, 5
(b) 2
(c) 3
(d) 5
(e) It prevents the rule from firing again.
5. To represent the degree of certainty the user has in an answer to a question, to represent the degree of certainty in the expert's knowledge (i.e. in a rule), to represent the degree of certainty in the conclusion.
6. There are many acceptable answers, including:
Driver: 'I think that there was oil on the road.'
Driver: 'I was driving at about 26 miles per hour.'
Driver: 'The rain was quite heavy.'

Chapter 10

Exercise 10.1

1.
 - (a) Analytical chemistry, classification, to identify the chemical structure of organic compounds from mass spectrometry.
 - (b) Robotic movement, planning, to plan a set of movements and actions for a robot in a given environment.
 - (c) Bacterial infections, classification/diagnosis, to identify bacterial blood infections in order to provide treatment for a patient.
 - (d) Mineral geology, advice, to recommend a location for drilling to locate mineral deposits.
 - (e) Cancer treatment, planning, to plan a set of treatments for a cancer patient.

2.
 - (a) Use of certainty factors to represent uncertainty in domain knowledge and conclusions.
 - (b) Use of automated knowledge acquisition to speed up the development and maintenance of the expert system.
 - (c) Use of working memory for forward chaining, development of RETE algorithm for maintaining a conflict set, use of conflict resolution strategies.

Chapter 11

Exercise 11.1

1.
 - (a) The company will save money by not doing trial excavations in useless areas.
 - (b) This will save land from being needlessly ruined.

2.
 - (a) Mistakes made by tired pilots will be avoided. The computer won't forget to take certain factors into account.
 - (b) Many examples exist of disasters being avoided because the pilot has carried out a manoeuvre that extended the technical feasibility of the aircraft. Fly-by-wire systems would not be able to make such manoeuvres because they can only perform actions that are within the range of permissible manoeuvres.
 - (c) Certainly, this question will keep the legal experts arguing for a while. If a specific error can be identified, then the finger of

guilt can be pointed at the guilty party with confidence. If no clear evidence emerges, then an argument in favour of blaming each of the parties listed can be made.

- (d) The system will ensure maximum fuel economy.
- (e) The system won't need additional cabling and controls for the pilot.

Difficulties associated with knowledge acquisition:
interviewing domain experts, understanding domain knowledge.

Also, need for extensive testing over a period of time.

3.
 - (a) This is very much a matter of personal opinion. Good justification can be provided for both sides of the argument.
 - (b) Many people have a serious objection to junk mail and dislike the idea of being targeted by companies because of their spending patterns. Other people may find it helpful to get information about items relating to what they generally buy.

4.
 - (a) Possible advantages: A doctor may have to consider a large number of facts and combinations of symptoms. If the patient has a rare disease, the doctor may miss a symptom but the expert system will identify it and provide a list of alternatives for the doctor. It will also be able to provide the results of the latest research, which the doctor may not have had time to read.
Possible disadvantages: It may slow the doctor down and therefore mean that fewer patients are seen. It may provide too many alternatives and lead to unnecessary tests.
 - (b) A new doctor will probably appreciate the extra confidence it provides.
An older doctor may be nervous of new technology.
 - (c) The doctor? The knowledge engineer? The domain expert?
Once again, justification can be provided for placing the blame with each party.
 - (d) No: the doctor can take account of common sense and human feelings.
 - (e) It may not be possible to have a fully trained doctor within reach of every community. An expert system could be used to determine whether or not a patient needs hospital treatment and can suggest which drugs to use.

SECTION 5

This section contains some sample expert systems to illustrate the features of expert systems in Section 3.

You will be provided with the expert system files for each exercise. Appendix A at the end of this section (on pages 166–80) contains the knowledge required to complete the practical exercises.

Exercises 1–11 cover material at Intermediate 2 level.

Exercises 12 and 13 cover material at Higher level.

Intermediate 2

Exercise 1

Making a decision about which magazine to buy is not easy – walk into any newsagents and you will see a huge variety on sale. An expert system has been created to help make this decision. The expert system asks questions about your likes and dislikes and then gives suitable advice, which depends on your answers. This expert system is called **Magazine**.

Use this expert system to solve each of the following problems. Write down the advice that is suggested by the expert system in each case. You should check your answers at the end of the exercise.

1. Your friend, Manpal, likes sport and has a particular interest in golf. However, he hates all water sports, including sailing. Which magazine should he buy?
2. It's your uncle's birthday! He doesn't get out much but enjoys puzzles and crosswords. What magazine should you buy for him?
3. Todd is interested in computers and regularly uses a Macintosh at school. What magazine would you recommend for him to buy?
4. Camilla has joined the local photography club and would like to place an order for a magazine that would provide her with more information about photography. What magazine should she order?
5. Marcus McDuff, the local landowner, has loads of money to throw around. So much money that he has just hired a new butler called Billy. Billy is sent to the newsagents to buy a magazine for Marcus. What would you recommend?
6. Katy enjoys sport. She is a supporter of the local football team and is a member of the sailing club. Which magazines may be of interest to her?
7. Roswana is a crossword fanatic. She has a four-year-old daughter who still believes in Santa Claus. Which magazines should she buy?

Exercise 2

The expert system that you are about to use has limited knowledge to advise you on the selection of a video in a video rental shop. The expert system is called **Video**.

Use this expert system to find solutions to each of the problems described below. Write down the advice that is suggested by the expert system in each case. You should check your answers at the end of the exercise.

1. Sally is going round to her friend's house and she wants to take a funny video that they can both watch. What video should she hire?
2. John and Sahid are both 15 and go to the video shop to rent a science-fiction video. Which video should they hire?
3. Students at school have been told that they can watch a video at the end of term provided that it has some historical content. Which videos would be appropriate for them?
4. Jack and Jamie are both 19 and in the same college course. They have decided to save money and spend the night watching science-fiction videos. What choice do they have?
5. Your neighbour, Mrs Holmes, has broken her ankle and asked you to get a video for her to watch this evening. She likes romantic thrillers. Which video would be suitable for her?

Justification features of an expert system

All expert systems are able to justify the questions they ask and the advice they suggest. The justification features available are:

- 'How'
- 'Why'.

The 'Why' feature can be used during the consultation. Whenever the expert system asks the user a question, the user can respond by asking the expert system to explain *why* that question was asked.

The 'How' feature can be used once advice has been offered by the expert system. Whenever an expert system offers a piece of advice, the user can ask the expert system to explain *how* it arrived at that piece of advice.

The use of the justification features of an expert system allows the expert system to explain its questioning and give reasons for its advice in a similar way to a human expert.

You should now be able to make use of the justification features whenever you use an expert system to solve problems.

Exercise 3

The video advisor gave advice on which video to rent from the video shop. Use the justification features of this expert system to answer each of the following questions. Write down the justification given in each case and check your answers at the end of the exercise.

1. You are babysitting your five-year-old neighbour this evening and you would like to rent a video that will keep you both entertained. Why does the expert system ask you if will be watching the video with younger friends?
2. John and Sahid are both 15 years old and go to the video shop to rent a science-fiction video.
 - (a) What explanation does the expert system give for asking whether or not they like science fiction?
 - (b) Later in the consultation, the expert system asks if they are over 15 years old. What reason does it give for asking this?
 - (c) How does the expert system explain asking whether or not the viewers are at least 18 years old?
3. What reason(s) does this expert system give for asking:
 - (a) whether or not the viewer likes romantic thrillers?
 - (b) whether or not the viewer wants a film that has a historical content?
 - (c) whether or not the viewer wants to watch a sad movie?

Exercise 4

The magazine selector expert system gave advice on which magazine to buy. Use the justification features of this expert system to answer each of the following questions. Write down the justification given in each case and check your answers at the end of the exercise.

1. Your neighbour, Mrs Barr, would like to buy a new car and asks for advice. The advice offered by the expert system is 'You should buy *Auto Trader*'. What justification does the expert system provide for offering this advice?
2. Your friend, Manpal, likes sport and has a particular interest in golf. However, he hates all water sports, including sailing. The advice offered by this expert system is 'You should buy *Golf Monthly*'. What explanation is given for offering this advice?
3. Valentina has asked Santa Claus to bring her a rocking horse at Christmas.
 - (a) What advice does the expert system give?
 - (b) How does it justify this advice?
4. Asam likes using computers and wants to learn more about them. At work, he uses a PC.
 - (a) Which magazine would be useful for him?
 - (b) What explanation does the expert system give for offering this particular piece of advice?
5. Claire Brown would like to hire a nanny to look after her children and she is wealthy enough to have money to spare.
 - (a) Which magazine should she buy?
 - (b) Ask the expert system to explain this advice. What reasons does it give?

More complex expert systems

Exercise 5

The expert system that you are about to use has limited knowledge to advise you on where you might spend your time on a Saturday afternoon. This expert system is more complex than those you have used previously. It is called **Saturday**.

Use this expert system to solve the problems described below. Write down the advice or reason given in each case. Check your answers at the end of the exercise.

1. Today is Saturday and Kenneth is wondering where he can go. It's raining outside and he has spent all his money.
2. Sally and Katy have just been paid and would like some advice on where to go on a nice bright dry Saturday.
 - (a) What does the expert system suggest?
 - (b) What reason did the expert system give for asking whether or not the girls had any money?
3. It is Saturday and it is snowing. Your neighbour's son has a French exchange student staying with him. They don't want to spend too much money today because they have a trip planned for the following week. They ask you for advice on where to go. What would you suggest?
4. Where can Angus take his girlfriend on a wet Saturday afternoon? He has just been paid for the week so money is no object.
5. What explanation does this expert system give for asking what the weather is like?
6. What explanation does this expert system give for offering the advice 'You should go shopping at the outdoor market'?
7. What advice is given to Ian and Ravinder on how to spend their Saturday afternoon? It is dry outside but they are feeling rather miserable as they have spent all their grant.
8. Lynn and Lorraine are wondering how to spend their Saturday. It is a lovely snowy day and they have just been paid so they don't mind splashing out.

Exercise 6

What do you do when a child's favourite toy breaks? Can it be repaired and if so what type of repair is necessary? An expert system has been written to help with such problems. It is called **Toys**.

Use this expert system to solve the problems described below. Write down the advice or reasons suggested in each case and check your answers at the end of the exercise.

1. Young Sam has a squeaky mouse. Normally, when Sam squeezes its tummy, the mouse squeaks. At the moment, the mouse won't squeak. What should be done to fix the mouse?
2. Jacinda has a battery-operated dog. Normally, the dog jumps up and down and barks but something is wrong – the dog won't bark. What should be done?
3. Your young niece, Rebecca, has a wind-up frog. When the frog is wound up, it hops across the room. But something is wrong – it won't move.
 - (a) What advice does the expert system give about fixing the frog?
 - (b) What justification does the expert system give for offering this advice?
4. Jason has a battery-operated clown that tells jokes. Jason is playing with the clown and has discovered that the clown is no longer telling jokes. He comes running to you for help.
 - (a) How does the expert system justify asking what type of toy is faulty?
 - (b) What advice does the expert system give in this case?
5. The local leisure pool has inflatable water wings for children to use when they come for a swim. Some of the water wings keep deflating because the air is escaping from them. What should be done to fix them?

Problem-solving practice

In this section, you will gain some further practice in using expert systems before attempting the assessment tasks.

Exercise 7

Much of a doctor's time is spent diagnosing common non-serious illnesses. It would make better use of doctors' expert knowledge if they could deal quickly with patients who are suffering from serious illnesses. An expert system has been written to help diagnose common non-serious illnesses so that doctors would have more time to deal with more serious cases. It is called **Illnesses**. Use this expert system to solve the problems described below.

1. Gamada is complaining of a sore throat and sore feet. What is he likely to be suffering from?
2. Kathy has a headache, a runny nose and a sore throat. She does not have a high temperature. What illnesses does the expert system suggest she could be suffering from?
3. Mulunish has arrived home complaining of having sore feet and no money. What is the likely cause of her symptoms?
4. Jim has a sore throat and a headache. He does not have a runny nose.
 - (a) What justification does the expert system give for asking if Jim has a high temperature?
 - (b) What advice does the expert system give?
5. Tony has been sent home from school with a headache and a bleeding nose. What is the likely cause of his symptoms?
6. Carol has gone to the medical room at work complaining of a sore throat. She does not have a headache.
 - (a) What advice is suggested by the expert system?
 - (b) What reasons does it give for suggesting this?

Exercise 8

An expert system has been written to help categorise different methods of transport. It is called **Transport**. Use this expert system to solve the problems described below.

1. A certain vehicle runs on rails, carries passengers and is powered by electricity.
 - (a) What different types of vehicle could this be?
 - (b) What justification does the expert system provide when offering the advice 'Means of transport is a train'?
2. Sally is describing a means of transport: 'It carries passengers, runs on the road and is powered by an engine.' What could she be describing?
3. On holiday, Houda sees a vehicle carrying passengers on water. The vehicle is pedal powered. What type of vehicle is she looking at?
4. Kenneth is driving a horse-driven vehicle on the road. The vehicle is used to transport cargo.
 - (a) The expert system asks if the vehicle is used to carry passengers or to transport cargo. What reason does it give for asking this?
 - (b) What type of vehicle is Kenneth driving?
5. What vehicles could be described as being used to carry passengers in the air and are powered by engines?

Exercise 9

A lot of careful preparation goes into a successful party. The location, catering and entertainment all have to be considered. An expert system has been written to help. It is called **Party**. Use this expert system to solve the problems described below.

1. Asam is planning a party to celebrate the end of his exams – he is 16. The party will have to be as cheap as possible. There is no need for a special cake or invitations. Asam would like help to plan the location, entertainment and the catering for the party. What advice would you give him?
2. Romeo and Juliet are planning a party to celebrate their engagement (both are over 18). They want the party to be as cheap as possible and they have decided to have it at Romeo's house. They want help in planning the entertainment, the cake, the invitations and the catering. What advice would you give them?
3. Rory's mum is planning his fifth birthday party and cost is not a problem. Rory would like a Micky Mouse birthday cake and matching invitations. His mum would like as much help as possible. What advice would you give her?
4. Johanna is about to celebrate her eighteenth birthday. Her parents have agreed to pay for a party with no expense spared. A special cake will be needed and invitations will have to be arranged. Johanna's mum is doing the catering herself. What advice would you give Johanna to help organise the entertainment, the location, the cake and the invitations?
5. Chris is organising a birthday party for his ten-year-old cousin, Craig. A special birthday cake is needed but invitations are not – the party goers will be contacted by phone instead. Chris has been saving up for the party so expense is not a problem. He needs help with the location, catering, entertainment and cake. What advice would you give?

Exercise 10

An expert system has been written to help plan and organise a successful holiday. The expert system provides advice on where to go, where to stay and what method of travel to use. The expert system is called **Holiday**. Use this expert system to solve the problems described below.

1. Mr and Mrs MacDonald have saved a substantial amount of money to take their two young children on holiday in the summer. They would prefer to do their own catering because the children are such fussy eaters. What advice does the expert system suggest?
2. Sisters, the two elderly Misses Took, wish to take a winter break somewhere in the UK. They don't have much money, so they have decided that self-catering would be a good way to keep the cost down. What advice does the expert system offer?
3. Four sixth-year pupils have decided to go on holiday together in the summer to celebrate leaving school. They want to go abroad for the sunshine but hope to keep the cost to a minimum by doing their own catering and travelling as cheaply as possible.
 - (a) What justification does the expert system give for asking whether or not the group want to go self-catering?
 - (b) What advice is offered by the expert system?
4. A couple would like to take their two teenaged children on a winter break somewhere in the UK. They want to relax as much as possible and wish to have their meals provided. What advice is suggested by the expert system?
5. Old Mr and Mrs Singh are hoping to go on holiday abroad to escape some of the severe winter weather. They are not short of money and would prefer not to do any cooking when they are on holiday.
 - (a) Why does the expert system ask if they have a lot of money to spend?
 - (b) What advice does the expert system give Mr and Mrs Singh?
 - (c) What justification does it give for offering this advice?

Exercise 11

Often, fish can be identified by their appearance. An expert system has been written to identify the type of fish being described. It is called **Fish**. Use this expert system to solve the problems described below.

1. Sally is describing the fish she has just caught. 'It has a forked tail and three fins but it doesn't have stripes'. What type of fish did she catch?
2. Kashia sees a fish in her local fishmonger's shop. It is striped and has a forked tail as well as two fins. What type of fish did she see?
3. The expert system asks whether or not the fish is stripy. What reason does it give for asking this question?
4. Andy catches a fish that has no stripes and a tail. It has two fins. What type of fish did Andy catch?
5. George is describing the fish that got away: 'It didn't have any stripes but it had a tail and three fins.' Which fish managed to get away?
6. What type of stripy fish has a forked tail and three fins?
7. Young Harry, your five-year-old nephew, sees a fish at the aquarium. The fish has stripes, three fins and a tail. What type of fish did Harry see?

Higher

Exercise 12

The species expert system is a 'classic' system used for testing features of expert system shells. It classifies a small number of animal species according to observed characteristics such as body covering, colour, markings, motion, food, reproduction and some others.

1. Identify a creature that has tawny fur with dark spots, eats meat and has short legs and neck.
2. Identify a creature that has black and white fur, hoofs and eats grass.
3. Identify a creature that is black and white, can't swim or fly, and lays eggs.
4. Critically evaluate the species expert system in terms of the metrics identified in Chapter 7. In particular, you should be able to comment critically on the method of reasoning used in this system.

Exercise 13 (H)

1. Adapt and extend the expert system to include the following:

Creatures can live on land or in water. Creatures that can live both on land and in the water are called amphibians. Creatures that live in water can swim, and creatures that live on land can walk.

Insectivores are creatures that eat only insects. Omnivores are creatures that eat a range of food, such as meat, fish, insects and plants.

A duck-billed platypus is a mammal that lives in water, lays eggs and feeds its young on milk. It has brown fur.

A marsupial is a mammal that gives birth to live young, has a pouch, and feeds its young on milk. All marsupials are found in Australia. Examples of marsupials are koalas, which live in trees, and kangaroos, which have short front legs and large hind legs, with a long tail.

Whales and dolphins are examples of mammals that live in water. Whales eat plankton, whereas dolphins eat fish. Seals are mammals that live in water and on land. They eat fish.

Fish have a scaly body covering and lay eggs. They live in water and breathe using gills. They eat mainly insects, although some eat fish or meat. Minnows are fish that eat insects; carp eat insects and fish; sharks eat fish and meat.

Reptiles have a scaly body covering and lay eggs. They use lungs for breathing. Lizards are four-legged, live on land and eat insects. Snakes are carnivorous, have no legs and live on land.

Create a factor table or decision tree to represent the information above. You should make use of existing conditions (such as family = mammal, motion = swims, eats = meat). Take care to structure the knowledge carefully and refine your conditions as necessary. Your final system should *not* ask questions directly about whether the creature is a marsupial, fish or reptile!

2. Critically evaluate your refined expert system.

Appendix A

This section contains the knowledge required to complete the practical exercises in Section 5. These are in the form of factor tables and must be converted into expert system files in the chosen shell.

Exercise 1: Magazine selector

Output	Condition 1	Condition 2
You should buy <i>Auto Trader</i>	You want to buy a car	
You should buy <i>Playdays</i>	You still believe in Santa Claus	
You should buy <i>Photo Advice</i>	You enjoy taking photographs	
You should buy <i>MacUser</i>	You like computers	You often use a Macintosh
You should buy <i>PC World</i>	You like computers	You often use a PC
You should buy <i>Homes and Gardens</i>	You have a lot of money to spare	You have a butler
You should buy <i>Nursery World</i>	You have a lot of money to spare	You need a nanny to look after the children
You should buy <i>Match</i>	You enjoy sport	You are interested in football
You should buy <i>Golf Monthly</i>	You enjoy sport	You are interested in golf
You should buy <i>Yachting Life</i>	You enjoy sport	You are interested in sailing
You should buy <i>Puzzle Weekly</i>	You enjoy crosswords	
You should keep your magazines in a magazine rack		

Exercise 2: Video advisor

Output	Condition 1	Condition 2
Choose <i>Jungle Book</i>	You intend watching with younger friends	
Rent <i>Lethal Weapon</i>	A comedy film is desired	
Rent <i>Back to the Future 3</i>	you like science fiction	Viewers are at least 15 years old
Choose <i>Alien 3</i>	You like science fiction	Viewers are at least 18 years old
Rent <i>The Bodyguard</i>	You like romantic thrillers	
You can save £1 on video hire if you rent two videos at the same time		
Choose <i>A League of Their Own</i>	You want to watch a sad movie	
Rent <i>Robin Hood Prince of Thieves</i>	You require a film with historical content	
Rent <i>Dances with Wolves</i>	You require a film with historical content	
The video shop closes at 9pm		

Exercise 5: Saturday outing

Output	Condition 1	Condition 2
Go to the library or museum	The weather is wet	You have no money
Go to the cinema	The weather is wet	You have some money to spend
Go for a walk in the park	The weather is dry	You have no money
Go sledging with friends	The weather is snowy	You have no money
Go skiing at the Glencoe Ski Centre	The weather is snowy	You have some money to spend
Go shopping at the outdoor market	The weather is dry	You have some money to spend
Always check the weather before leaving the house		

Exercise 6: Toy fault diagnosis

Output	Condition 1	Condition 2
Buy a replacement squeaker from the manufacturer	Toy can be squeezed	Toy will not squeak
Buy a replacement squeaker from the manufacturer	Toy can be squeezed	Toy will not make a noise
Buy replacement batteries	Toy is battery operated	Toy is not moving
Buy replacement batteries	Toy is battery operated	Toy will not make a noise
Wind the winder	Toy has a winder	Toy is not moving
Use a puncture repair kit to repair the toy	Toy is inflatable	Air is escaping from the toy

Exercise 7: Illnesses

Output	Condition 1	Condition 2	Condition 3
It is likely that the patient has the flu	Patient has a high temperature	Patient has a headache	
It is likely that the patient has a cold	Patient has a runny nose	Patient has a headache	
It is likely that the patient has chickenpox	Patient has a high temperature	Patient has itchy spots	
It is likely that the patient has mumps	Patient has a high temperature	Patient has a swollen mouth	
It is likely that the patient has been in a fight	Patient has a headache	Patient has a bleeding nose	
It is likely that the patient has a throat infection	Patient has a headache	Patient has a sore throat	
It is likely that the patient has spent too much time at the sales	Patient has sore feet	Patient has no money left	
It is likely that the patient has spent too much time at a loud disco	Patient has sore feet	Patient has a sore throat	Patient doesn't have a headache (Use NOT)
It is likely that the patient likes the sound of their own voice and talks too much	Patient has a sore throat	Patient doesn't have a headache (Use NOT)	

Exercise 9: Transport

Output	Condition 1	Condition 2	Condition 3
Means of transport is a train	Vehicle travels on a rail	Vehicle carries passengers	Vehicle is powered by electricity
Means of transport is a truck	Vehicle travels on the road	Vehicle carries cargo	Vehicle is powered by an engine
Means of transport is a bus	Vehicle travels on the road	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is an aeroplane	Vehicle travels in the air	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is a ferry	Vehicle travels on water	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is a hot-air balloon	Vehicle travels in the air	Vehicle carries passengers	Vehicle is powered by hot air
Means of transport is a bicycle	Vehicle travels on the road	Vehicle carries passengers	Vehicle is powered by pedals
Means of transport is a sleigh	Vehicle travels on the road	Vehicle carries passengers	Vehicle is powered by horse
Means of transport is a train	Vehicle travels on a rail	Vehicle carries cargo	Vehicle is powered by electricity
Means of transport is an aeroplane	Vehicle travels in the air	Vehicle carries cargo	Vehicle is powered by an engine
Means of transport is a ferry	Vehicle travels on water	Vehicle carries cargo	Vehicle is powered by an engine
Means of transport is a tram	Vehicle travels on a rail	Vehicle carries passengers	Vehicle is powered by electricity
Means of transport is a taxi	Vehicle travels on the road	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is a car	Vehicle travels on the road	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is a cruise liner	Vehicle travels on water	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is a helicopter	Vehicle travels in the air	Vehicle carries passengers	Vehicle is powered by an engine
Means of transport is a cart	Vehicle travels on the road	Vehicle carries cargo	Vehicle is powered by horse
Means of transport is a pedalo	Vehicle travels on water	Vehicle carries passengers	Vehicle is powered by pedals
Means of transport is a lorry	Vehicle travels on the road	Vehicle carries cargo	Vehicle is powered by an engine

Exercise 9: Party planning – part 1

Output	Condition 1	Condition 2	Condition 3	Condition 4
A suitable location would be at home	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is under 8	Cost of party should be cheap
A suitable location would be a local hall	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is under 8	Cost of party should be expensive
A suitable location would be at home	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is 8 to 14	Cost of party should be cheap
A suitable location would be a cinema	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is 8 to 14	Cost of party should be expensive
A suitable location would be a local hall	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is 14 to 17	Cost of party should be cheap
A suitable location would be a cinema	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is 14 to 17	Cost of party should be expensive
A suitable location would be at home	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is adult	Cost of party should be cheap
A suitable location would be a local hall	You would like help with the location	Reason to celebrate is a birthday	Age of party-goers is adult	Cost of party should be expensive
A suitable location would be at home	You would like help with the location	Reason to celebrate is end of exams	Age of party-goers is 14 to 17	Cost of party should be cheap
A suitable location would be a local hall	You would like help with the location	Reason to celebrate is end of exams	Age of party-goers is 14 to 17	Cost of party should be expensive
A suitable location would be at home	You would like help with the location	Reason to celebrate is end of exams	Age of party-goers is adult	Cost of party should be cheap
A suitable location would be a local hall	You would like help with the location	Reason to celebrate is end of exams	Age of party-goers is adult	Cost of party should be expensive
A suitable location would be a local hall	You would like help with the location	Reason to celebrate is engagement or anniversary	Age of party-goers is adult	Cost of party should be cheap
A suitable location would be a hotel	You would like help with the location	Reason to celebrate is engagement or anniversary	Age of party-goers is adult	Cost of party should be expensive

Exercise 9: Party planning – part 2

Output	Condition 1	Condition 2	Condition 3	Condition 4
Suitable entertainment would be party games	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is under 8	Cost of party should be cheap
Suitable entertainment would be Cheeko the Clown	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is under 8	Cost of party should be expensive
Suitable entertainment would be videos	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is 8 to 14	Cost of party should be cheap
Suitable entertainment would be movie watching	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is 8 to 14	Cost of party should be expensive
Suitable entertainment would be music tapes	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is 14 to 17	Cost of party should be cheap
Suitable entertainment would be a disco	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is 14 to 17	Cost of party should be expensive
Suitable entertainment would be music tapes	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is adult	Cost of party should be cheap
Suitable entertainment would be a disco	You would like help with the entertainment	Reason to celebrate is a birthday	Age of party-goers is adult	Cost of party should be expensive
Suitable entertainment would be music tapes	You would like help with the entertainment	Reason to celebrate is end of the exams	Age of party-goers is 14 to 17	Cost of party should be cheap
Suitable entertainment would be a disco	You would like help with the entertainment	Reason to celebrate is end of exams	Age of party-goers is 14 to 17	Cost of party should be expensive
Suitable entertainment would be music tapes	You would like help with the entertainment	Reason to celebrate is end of exams	Age of party-goers is adult	Cost of party should be cheap
Suitable entertainment would be a disco	You would like help with the entertainment	Reason to celebrate is end of exams	Age of party-goers is adult	Cost of party should be expensive

Suitable entertainment would be music tapes	You would like help with the entertainment	Reason to celebrate is engagement or anniversary	Age of party-goers is adult	Cost of party should be cheap
Suitable entertainment would be a disco	You would like help with the entertainment	Reason to celebrate is engagement or anniversary	Age of party-goers is adult	Cost of party should be expensive

Exercise 9: Party planning – part 3

Output	Condition 1	Condition 2	Condition 3	Condition 4
Suitable food would be ice-cream and jelly	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is under 8	Cost of party should be cheap
Suitable food would be ice-cream and jelly	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is under 8	Cost of party should be expensive
Suitable food would be sandwiches	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is 8 to 14	Cost of party should be cheap
Suitable food would be burgers	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is 8 to 14	Cost of party should be expensive
Suitable food would be a home-made buffet	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is 14 to 17	Cost of party should be cheap
Suitable food would be burgers	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is 14 to 17	Cost of party should be expensive
Suitable food would be a home-made buffet	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is adult	Cost of party should be cheap
Suitable food would be a bought buffet	You would like help with the food	Reason to celebrate is a birthday	Age of party-goers is adult	Cost of party should be expensive
Suitable food would be a home-made buffet	You would like help with the food	Reason to celebrate is end of the exams	Age of party-goers is 14 to 17	Cost of party should be cheap
Suitable food would be a bought buffet	You would like help with the food	Reason to celebrate is end of exams	Age of party-goers is 14 to 17	Cost of party should be expensive
Suitable food would be a home-made buffet	You would like help with the food	Reason to celebrate is end of exams	Age of party-goers is adult	Cost of party should be cheap
Suitable food would be a bought buffet	You would like help with the food	Reason to celebrate is end of exams	Age of party-goers is adult	Cost of party should be expensive
Suitable food would be a buffet	You would like help with the food	Reason to celebrate is engagement or anniversary	Age of party-goers is adult	Cost of party should be cheap
Suitable food would be a sit-down meal	You would like help with the food	Reason to celebrate is engagement or anniversary	Age of party-goers is adult	Cost of party should be expensive

Exercise 9: Party planning – part 4

Output	Condition 1	Condition 2	Condition 3
Arrange for a celebration cake to be made at home	You would like help with celebration cake	Celebration cake is required	Cost of party should be cheap
Arrange for a celebration cake to be bought	You would like help with celebration cake	Celebration cake is required	Cost of party should be expensive

Exercise 9: Party planning – part 5

Output	Condition 1	Condition 2	Condition 3
Arrange for invitations to be made at home	You would like help with the invitations	Invitations are needed	Cost of party should be cheap
Arrange for invitations to be bought	You would like help with the invitations	Invitations are needed	Cost of party should be expensive
Arrange for invitations to be written and sent out	You would like help with the invitations	Invitations are needed	
Send invitations to school friends of playmates	You would like help with the invitations	Reason to celebrate is a birthday	Age of party-goers is under 8
Send invitations to school friends	You would like help with the invitations	Reason to celebrate is a birthday	Age of party-goers is 8 to 14
Send invitations to school friends	You would like help with the invitations	Reason to celebrate is a birthday	Age of party-goers is 14 to 17
Send invitations to friends and colleagues	You would like help with the invitations	Reason to celebrate is a birthday	Age of party-goers is adult
Send invitations to school friends	You would like help with the invitations	Reason to celebrate is end of exams	Age of party-goers is 14 to 17
Send invitations to friends and colleagues	You would like help with the invitations	Reason to celebrate is end of exams	Age of party-goers is adult
Send invitations to friends, colleagues and relations	You would like help with the invitations	Reason to celebrate is engagement or anniversary	Age of party-goers is adult

Exercise 10: Holiday planner – part 1

Output	Condition 1	Condition 2	Condition 3	Condition 4
A suitable location would be Largs	You would like help with the location	Season for holiday is summer	Age of holiday-goers is pensioners	Holiday to be taken in the UK
A suitable location would be Malta	You would like help with the location	Season for holiday is summer	Age of holiday-goers is pensioners	Holiday to be taken abroad
A suitable location would be Brighton	You would like help with the location	Season for holiday is winter	Age of holiday-goers is pensioners	Holiday to be taken in the UK
A suitable location would be Granada and Seville	You would like help with the location	Season for holiday is winter	Age of holiday-goers is pensioners	Holiday to be taken abroad
A suitable location would be Blackpool	You would like help with the location	Season for holiday is summer	Age of holiday-goers is couple with teenagers	Holiday to be taken in the UK
A suitable location would be the South of France	You would like help with the location	Season for holiday is summer	Age of holiday-goers is couple with teenagers	Holiday to be taken abroad
A suitable location would be Centre Parks	You would like help with the location	Season for holiday is winter	Age of holiday-goers is couple with teenagers	Holiday to be taken in the UK
A suitable location would be an Austrian ski resort	You would like help with the location	Season for holiday is winter	Age of holiday-goers is couple with teenagers	Holiday to be taken abroad
A suitable location would be Butlins	You would like help with the location	Season for holiday is summer	Age of holiday-goers is couple with young children	Holiday to be taken in the UK
A suitable location would be Alcudia	You would like help with the location	Season for holiday is summer	Age of holiday-goers is couple with young children	Holiday to be taken abroad
A suitable location would be Bournemouth	You would like help with the location	Season for holiday is winter	Age of holiday-goers is couple with young children	Holiday to be taken in the UK
A suitable location would be Disneyland Paris	You would like help with the location	Season for holiday is winter	Age of holiday-goers is couple with young children	Holiday to be taken abroad
A suitable location would be Devon	You would like help with the location	Season for holiday is summer	Age of holiday-goers is young adults	Holiday to be taken in the UK
A suitable location would be the Greek Islands	You would like help with the location	Season for holiday is summer	Age of holiday-goers is young adults	Holiday to be taken abroad
A suitable location would be Aviemore	You would like help with the location	Season for holiday is winter	Age of holiday-goers is young adults	Holiday to be taken in the UK
A suitable location would be Teneriffe	You would like help with the location	Season for holiday is winter	Age of holiday-goers is adults	Holiday to be taken abroad

Exercise 10: Holiday planner – part 2

Output	Condition 1	Condition 2	Condition 3	Condition 4
Suitable accommodation would be a guest house	You would like help with accommodation	Age of holiday-goers is pensioners	Holiday to be taken in the UK	Sit-down meals are required
Suitable accommodation would be a holiday flat	You would like help with accommodation	Age of holiday-goers is pensioners	Holiday to be taken in the UK	Self-catering is required
Suitable accommodation would be a studio flat	You would like help with accommodation	Age of holiday-goers is pensioners	Holiday to be taken abroad	Self-catering is required
Suitable accommodation would be a hotel	You would like help with accommodation	Age of holiday-goers is pensioners	Holiday to be taken abroad	Sit-down meals are required
Suitable accommodation would be a guest house	You would like help with accommodation	Age of holiday-goers is couple with teenagers	Holiday to be taken in the UK	Sit-down meals are required
Suitable accommodation would be a holiday flat	You would like help with accommodation	Age of holiday-goers is couple with teenagers	Holiday to be taken in the UK	Self-catering is required
Suitable accommodation would be a hotel	You would like help with accommodation	Age of holiday-goers is couple with teenagers	Holiday to be taken abroad	Sit-down meals are required
Suitable accommodation would be a caravan	You would like help with accommodation	Age of holiday-goers is couple with teenagers	Holiday to be taken abroad	Self-catering is required
Suitable accommodation would be a guest house	You would like help with accommodation	Age of holiday-goers is couple with young children	Holiday to be taken in the UK	Sit-down meals are required
Suitable accommodation would be a caravan or cottage	You would like help with accommodation	Age of holiday-goers is couple with young children	Holiday to be taken in the UK	Self-catering is required
Suitable accommodation would be a hotel	You would like help with accommodation	Age of holiday-goers is couple with young children	Holiday to be taken abroad	Sit-down meals are required
Suitable accommodation would be a villa with swimming pool	You would like help with accommodation	Age of holiday-goers is couple with young children	Holiday to be taken abroad	Self-catering is required
Suitable accommodation would be a small hotel	You would like help with accommodation	Age of holiday-goers is young adults	Holiday to be taken in the UK	Sit-down meals are required
Suitable accommodation would be cottage or holiday flat	You would like help with accommodation	Age of holiday-goers is young adults	Holiday to be taken in the UK	Self-catering is required
Suitable accommodation would be a flat or villa	You would like help with accommodation	Age of holiday-goers is young adults	Holiday to be taken abroad	Self-catering is required
Suitable accommodation would be a hotel	You would like help with accommodation	Age of holiday-goers is young adults	Holiday to be taken abroad	Sit-down meals are required

Exercise 10: Holiday planner – part 3

Output	Condition 1	Condition 2	Condition 3
Travel by own car or hired car	You would like help with transport	Cost should be as low as possible	Holiday to be taken in the UK
Travel by luxury coach or train	You would like help with transport	Cost is not a problem	Holiday to be taken in the UK
Travel by own car or luxury coach	You would like help with transport	Cost should be as low as possible	Holiday to be taken abroad
Travel by plane	You would like help with transport	Cost is not a problem	Holiday to be taken abroad

Exercise 11: Fish identifier

Output	Condition 1	Condition 2	Condition 3
Fish is a snatcher	Fish is striped	Fish has a forked tail	
Fish is an addend	Fish is not striped	Fish has a forked tail	Fish has fins
Fish is a thwait	Fish is not striped	Fish does not have a forked tail	Fish does not have fins
Fish is a sea-flyer	Fish is striped	Fish does not have a forked tail	
Fish is a bracketer	Fish is striped	Fish has a forked tail	Fish does not have fins
Fish is a kiro	Fish is striped	Fish has a forked tail	Fish has fins

Exercise 12: Species classification

Species is cheetah

IF feeding type is carnivore
AND colour is tawny
AND marking is dark_spots
AND legs_and_neck is short.

Species is tiger

IF feeding type is carnivore
AND colour is tawny
AND marking is black_stripes.

Species is giraffe

IF feeding type is ungulate
AND colour is tawny
AND marking is dark_spots
AND legs_and_neck is long.

Species is zebra

IF feeding type is ungulate
AND colour is black_and_white.

Species is ostrich

IF family is bird
AND motion is walks
AND colour is black_and_white
AND legs_and_neck is long.

Species is penguin

IF family is bird
AND motion is swims
AND colour is black_and_white.

Species is albatross

IF family is bird.

Family is mammal

IF body_covering is hair
OR (body_covering is other
AND feeds_young_on is milk).

Family is bird

IF body_covering is feathers
OR (motion is flies
AND the reproduction is eggs).

Feeding type is carnivore

IF family is mammal
AND (eats is meat)
OR (teeth is pointed
AND feet is claws
AND eyes is point_forward).

Feeding type is ungulate

IF family is mammal
AND (eats is grass
OR feet is hoofs).