Complete Computer Science for Cambridge IGCSE® & O Level

Alison Page
David Waters
Introduction

If you are studying Computer Science for Cambridge IGCSE® or O Level, then this book is designed for you. Its purpose is to help you achieve your best in the course and examination, equipping you with the knowledge you need to study the subject at a higher level.

The book follows the latest syllabuses and consists of ten chapters: chapters 1-7 cover the first section of the syllabus (Theory of Computer Science), while chapters 8-10 cover the second section (Problem-solving and programming).

Each chapter consists of topics that map to the sub-sections of the syllabus; each topic consists of several two-page units called spreads. To help you make the most of this student book, the following features are used to organise the content:

**Syllabus reference**
The explicit syllabus reference allows you to frame your learning and build connections between different topics.

**Introduction**
The short introduction will make it clear what you will learn in each lesson.

The material marked with a line on the side goes beyond the requirements of the syllabus and will not be tested in the examination. It’s intended to give you a broader understanding of computer science and hopefully you’ll find it interesting!

**Test yourself & Learning activity**
The “Test yourself” and “Learning activity” questions will help you check your understanding after each lesson and give you further opportunities to practise what you have learnt.

**Key terms and interesting facts**
These key terms, interesting facts and tips will extend your understanding of the subject.

The review page at the end of each chapter highlights the new key terms you’ve learnt in that chapter and proposes further project work.

**Programming**
In this student book, Python is used as an example programming language. There are many other programming languages and your teacher might have chosen a different one. However, the features you will learn about are found in almost all languages.
# Software

## 5.1 Systems software
What is software?  
Operating systems  
Functions of an operating system

## 5.2 Computer languages
Low-level languages  
High-level languages  
Review

# Security

## 6.1 Security threats
Data security  
Security threats  
Malpractice and crime  
Online attacks

## 6.2 Security protection
Proof of identity  
Firewalls  
Security protocols  
Encryption  
Security examples  
Review

# Ethics

## 7.1 Ethics
Copyright  
Free software  
Hackers and crackers  
Review

# Programming

## 8.1 Introduction to programming
Introduction to Python  
Algorithms

## 8.2 Begin coding
Output  
Sequence  
Input  
Assign a value  
Calculated values  
Variables in pseudocode  
Variables in flowcharts

## 8.3 Selection
Logical decision  
Python `if...else...`

# Databases

## 10.1 Database design
Records and fields  
Data types  
Primary key

## 10.2 Database queries
Select fields  
Select records  
Review  
Index  
Answers  
www.oxfordsecondary.com/9780198367215
What’s on your **kerboodle**?

The *Complete Computer Science for Cambridge IGCSE and O Level Kerboodle* is an online learning platform, specifically designed to accompany this student book. If your school has a subscription, you will be able to access a bank of resources to support your learning, help you prepare for the examination and achieve your best.

**Your resources**

The *Complete Computer Science for Cambridge IGCSE and O Level Kerboodle* consists of three main modules:

1. The Lessons module contains lesson presentations and is a teacher-only area.
2. The Resources module is where you will find hundreds of worksheets and interactive activities to help you practise and develop your knowledge and skills.
3. The Assessment module is where you will see all the quizzes assigned to you by your teacher.

**On Your Marks**

The interactive **On Your Marks** are designed to help you build your skills and confidence when answering examination questions. Covering a range of question types, there is one activity for each of the topics covered in the student book. Each activity is split into two separate parts, allowing you to take a step-by-step approach:

- **Understand and prepare** (found in the Resources module of Kerboodle) gives you the opportunity to analyse an examination question and three sample student answers. It will then ask you to look at examiner feedback for those answers. Finally, you will have to give each sample answer a mark.

- **Test** (found in the Assessment module of Kerboodle, only available once your teacher assigns it to you) allows you to answer the question for yourself, and receive feedback and a mark from your teacher.
What's on your Kerboodle?

**WebQuest**
The WebQuest mini projects allow you to research specific aspects of the topics covered in the student book. Besides strengthening your understanding of the subject, they will help you develop skills such as communication and teamwork.

**Interactive activities**
There is a starter activity for each spread of the student book, specifically designed to consolidate your understanding of key concepts in computer science.

**Worksheets**
There is an extension worksheet for each spread of the student book, to extend your understanding and inspire further study of the subject.

---

### 4.3 Using storage devices and media

**Introduction**
Storage devices form a category of computer hardware. This WebQuest is concerned with secondary storage devices, which are ones that allow you to make more permanent copies of your files and data. Each type of storage device has different strengths and weaknesses which makes it more suited to specific purposes. Not every storage device is suitable for every task.

What storage devices are there and how do they work? What is the difference between a storage device and storage medium, and which medium goes with which device? For which purposes are these storage systems best suited to the real world?

**Task**
Do some research to find out about a range of storage devices and media. Some examples are hard disk drives (fixed and removable), solid state drives, DVD, CD, Blue-ray disc and USB flash memory. Your teacher may ask each group to research a different category of storage device.

First, find out how each device works, and if appropriate, what storage medium it uses. Try to discover the main benefits and drawbacks of each type of storage device and medium. You will then describe how each device is used in real life scenarios.

Create a presentation of about five minutes’ duration for the rest of your class. Your presentation should be informative, concise and include pictures of the devices and, if relevant, their associated media.

**Process**

#### Step 1: Roles
A useful way to organise this task would be for each group to be allocated a category of storage device and medium to research. Your group should decide how to tackle the task you have been assigned. One method would be for each member of the group to initially look at a particular type of storage device and medium, and where it may be used, gathering as much useful information as possible. Once this is done, the group as a whole will create the final presentation.

#### Step 2: Research
Carry out the research on your specific storage device(s) and any associated media, and find as much as you can about how it works, and its good and bad points. Make sure you have a picture of it. Then find out and explain how and where it can be used in real life situations.

---

### 4.1 Sensors

A Building Management System, or BMS, is a computerised system that works in a building to monitor and control major parts of that building’s environment. Some of the things a BMS controls are:

- temperature and ventilation
- lighting
- electrical power
- fire systems
- access systems (for example, entry and exit doors)

The top floor of a school needs to be brought into the Building Management System. You have been asked to plan where sensors will be installed. This is what you have been asked to do:

1. temperature-controlled sensors in every room
2. smoke sensors in all rooms plus three in the corridor to link into fire alarm system
3. motion sensors in every room to control lights and link into security alarm systems
4. light sensors in every room to close blinds automatically in direct sunlight

The money available for this work is £4,750. The table below shows the cost of installing each type of sensor and what it will cost if you must buy four of the requirements above in full. You will see straight away that you will not have enough money.

<table>
<thead>
<tr>
<th>Sensor Type</th>
<th>Cost per Sensor</th>
<th>Number of Sensors</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>£120</td>
<td>7</td>
<td>£840</td>
</tr>
<tr>
<td>Smoke</td>
<td>£270</td>
<td>10</td>
<td>£270</td>
</tr>
<tr>
<td>Motion</td>
<td>£170</td>
<td>7</td>
<td>£1,190</td>
</tr>
<tr>
<td>Light</td>
<td>£200</td>
<td>7</td>
<td>£1,400</td>
</tr>
</tbody>
</table>

**Total**

| Total Cost | £4,810 |

**Total Budget**

1. Suggest how the number of sensors can be reduced so that the cost is equal to or less than £4,750. For each of the four types of sensor, think about where they are really needed. For example, does the school really need to maintain a constant temperature in a store room? Do they need to have automatic light control in rooms that are in constant use? Use the plan below to help you think about this problem, and use the table above to record your suggestions.

Here: The compass might help you make some savings!
Java and Visual Basic programming support
If you are learning to program in Java or Visual Basic rather than Python, these PDF booklets will bring you all the material on the programming spreads in the student book, but with specific Java or Visual Basic syntax and examples.

Summative quizzes
One for each topic, these multiple choice quizzes allow you to test your knowledge and identify any areas of improvement.

Exam preparation
These exam-style practice papers will test your understanding of theory (Paper 1) and assess your problem-solving and programming skills (Paper 2 and pre-release materials).
1 Data representation
**Binary data**

**Introduction**

In this section you will learn what a computer does. You will learn why computers use binary data.

**What is a computer?**

A computer is an electronic machine for processing data. “Data” is a general term for facts and figures. All types of data can be processed by a computer. Computers can work with numbers, words, images, sounds, and video, and they can control physical processes. All of these types of data can be stored inside a computer. The computer processes the data. That means it turns the data into something more useful. The useful output of the computer is often called “information”.

Data is processed to make useful information

A computer processes data to make useful information. Computers are made by people, to be useful to people. There is no point in using a computer unless it does something you want.

**Inside the computer**

Inside every computer is a processor. The processor stores and processes data. The processor stores the data using electrical switches that can either be on or off. A switch that is on will conduct electricity. A switch that is off will not carry an electric signal.

“Binary” means anything that can be in one of two different states. A switch that can be on or off is binary. There are exactly two choices – no more, no less. The data inside a computer is stored in a binary format.

A flashlight can be in one of two states – either on or off
A computer can process data of many different kinds, but all types of data must first be turned into binary data so the computer can use it. That is why this chapter – the first in the book – teaches you about binary data.

### Representing binary data

When we want to show binary data we write it as a “binary number”: that is, a number made of 1s and 0s. We can also call this “base 2”. For example:

```
0 1 0 0 1 1 1 0
```

Of course, if you opened up a computer and looked in the processor you would not see 1s and 0s. That is just a convenient way of representing the on and off electrical signals.

---

**Test yourself**

Use the information on these pages to answer the following questions.

1. The computer does not have 1s and 0s inside its memory. Why do we sometimes represent computer data in that form?

2. Explain in your own words why data must be converted into binary form when it is input to a computer.

3. On this page a flashlight is given as an example of something that is binary (has two different states). Think of another example, from outside computer science.

4. Why are computers useful?

---

**Learning activity**

On this page, five different types of data are mentioned: numbers, words, images, sounds and video. Search on the Internet for a short example of each type of data. Store a sample of each type of data in your own storage area of the computer system.
Syllabus reference

1.1.1 Binary systems

Learners should be able to: show understanding of the concept of a byte and how the byte is used to measure memory size; use binary in computer registers for a given application.

See also:
1.3 Data storage
3.3 Inside the CPU

Bits and bytes

Introduction

You have learned that data is held in the computer in binary form. In this section you will learn how the size of computer memory is measured as the amount of binary data it can hold.

Bits and bytes

Binary numbers are made up of the digits 1 and 0 (one and zero). We abbreviate the term “binary digit” to “bit”. Therefore, in computer science, one bit is a 0 or a 1. One bit is the smallest piece of computer memory. A bit represents a single on/off switch inside the computer's electronic memory.

Inside the computer, on/off bits are organised into groups of eight. A group of eight bits is called a “byte”.

For this reason we usually write binary numbers in groups of eight bits. That matches the way a computer organises bits. If the number is less than eight bits long, we put extra 0s on the left of the digits to make it up to 8 bits. For example, the binary number 1 1 0 1 would be represented by this byte:

0 0 0 0 1 1 0 1

Measuring memory

The area of the computer processor that stores data as on/off electrical signals is called the memory (random-access memory or RAM). We measure the size of RAM by how many bytes it can hold.

This old advert boasts that a memory card will add 16K (kilobytes) to your computer’s RAM.
We measure memory in kilobytes, megabytes and gigabytes:

- A kilobyte is 1024 bytes. That will store about half a page of text.
- A megabyte is 1024 kilobytes. A picture may be 3 megabytes.
- A gigabyte is 1024 megabytes. A movie may be 4 gigabytes.

When computers are advertised they often say how many bytes of RAM there are in the computer processor.

**RAM and Computer Speed**

The computer processor makes changes to the data in RAM. That is called processing the data. It is very quick and easy for the processor to access the contents of RAM. The processing will be quick.

If there isn’t enough room in RAM, then some of the data must be stored outside of RAM. It is slower for the processor to access this data. That means the computer will go more slowly.

Adding more RAM to a computer will help it process data more quickly.

**Registers**

There may be 8 gigabytes of RAM in the main memory of a computer.

Processing does not happen in the main memory though. Data that is ready to be processed is copied into a much smaller area of memory called a register. Registers are normally measured by the number of bits they can hold, for example an 8-bit register or a 32-bit register.

A processor often contains several kinds of registers, each used for a different purpose. Computers with large registers can work on more data at a time than those with small registers, so they usually work more quickly.

**Test yourself**

1. Explain the difference between a bit and a byte.
2. Work out exactly how many bytes there are in a megabyte.
3. Roughly how many pictures could be stored using 1 gigabyte of data?
4. Explain two factors that increase the speed of a computer.

**Learning activity**

Using magazines and the Internet, collect adverts for computers currently for sale. What does each advert say about the computer? How big is the RAM?
Binary and denary

**Introduction**

You have learned that the computer stores data as binary numbers. In this section you will compare binary to denary (our normal number system).

**The denary system**

The number system that you are familiar with in everyday life uses ten different digits. We can also call this “base 10”. Every number that exists can be represented using these ten digits:

0 1 2 3 4 5 6 7 8 9

Something that can be in ten different forms is known as denary. Our number system uses ten different digits, so it is a denary system. Another name for it is ‘decimal’. It can also be referred to as “base 10”.

How do we represent every number, using just ten digits? The answer is we use the position of a digit to change its meaning. Look at these numbers:

702
720
207

Each of these denary numbers uses the digit 2. In the first case it means two. In the next example it means twenty. In the final example it means two hundred. The difference is the position of the digit in the number.

Start at the right of a number. The digit on the right stands for single units. As we move to the left each digit has a value that is ten times bigger.

<table>
<thead>
<tr>
<th>1000 (thousands)</th>
<th>100 (hundreds)</th>
<th>10 (tens)</th>
<th>1 (units)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

In this example we have seven hundreds, two tens and no units. That is the meaning of 720 in denary.

**The binary system**

Binary numbers are made of the binary digits 1 and 0 (also called bits). Every number that exists can be represented using just 1s and 0s.

As with the denary system, the position of the bit tells you its value. Start at the right of a number. The bit on the right stands for single units. As we move to the left each bit has a value that is two times bigger. Here is an example: the binary number 1 0 1.

<table>
<thead>
<tr>
<th>8 (eights)</th>
<th>4 (fours)</th>
<th>2 (twos)</th>
<th>1 (units)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

In this example we have one four, no twos and one unit. Adding the values together makes 5. Therefore, 1 0 1 in binary is 5 in denary.
Invention of the denary system
The denary number system that we use today was developed in India, probably in the 3rd century BCE (2,300 years ago). It was better for doing complex maths than any other number system that existed at that time.
This number system was adopted in the Arabic world, and spread beyond India. Around 850 CE, the Persian mathematician Al-Khwarizmi and the Arab mathematician Al-Kindi wrote books that explained the denary number system.
Later, Europeans learned this system from Arabic mathematicians. In Europe, denary digits are still sometimes called ‘Arabic numerals’.

Invention of the binary system
Binary number systems were developed much later. A German philosopher called Gottfried Leibniz set out the idea of binary numbers, made with 1s and 0s, in 1679. In the 19th century, a British mathematician called Ada Lovelace wrote notes about how binary numbers could be used in digital processing.

Test yourself
1. Why is our normal number system called a denary system?
2. Explain why a digit such as 9 can hold many different values in a denary number.
3. How many bits are there in the binary number 0010?
4. In the binary number 0010 what value does the 1 stand for?

Learning activity
Complete this table by replacing the question marks with number values. Remember each value is twice as big as the one before.

<table>
<thead>
<tr>
<th>?</th>
<th>?</th>
<th>?</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Denary
Anything that can exist in ten different states is known as denary. Our usual number system is denary.

Talking about binary numbers
When you talk about binary numbers, never use denary words. For example, the binary number 1 0 1 is not “one hundred and one”. It is “one zero one” or “one ‘oh’ one”.
Counting in binary

Introduction

You have learned about binary and denary numbers. In this section you learn how to count in binary, and how it compares to counting in ordinary (denary) numbers.

Counting in denary

To count up from zero in ordinary, denary, numbers you start at 0. You count in the “units” column. The digit in the “units” column gets 1 bigger each time:

00
01
02

When you get to 9, you have run out of digits:

09

How can you add more? You return the “units” column back to 0, and add 1 to the next column along. In denary that is the “tens” column:

10

You then start again, counting upwards in the “units” column. When you reach 9:

19

You reset the units column to 0 and add 1 in the “tens” column:

20

In this way you can count up to any number using denary.

Counting in binary

When you count in binary it is the same. Start with 0:

0

Then add 1:

1

You have used 0 and 1 and there are no more binary digits. Therefore, you return the “units” column to 0, and add 1 in the column to the left:

10

Continue to count. Add 1 in the “unit” column:

11

Now you have run out of digits in both columns, so you reset them both to 0, and add a 1 in the column to the left:

100

In this way you can count up to any number in binary. However, the numbers can get quite long.
Group learning activity

Form a group of eight students. Sit in a row on eight chairs. Each student will be one of the bits in a byte. The rest of the class can watch. Later, everyone will get a turn.

The teacher is going to count up in the normal way. The students are going to act out binary counting.

- A student who is sitting down means 0.
- A student who is standing up means 1.

The chair on the right of the line represents the “units” column. The student in this chair has the most work to do. Every time the teacher counts, the student in this chair has to change position. If you are that student, you have to follow these rules:

- If you are sitting down, stand up.
- If you are standing up, turn and nod to the person next to you, and sit down.

The students in the other chairs don’t move. They wait and watch the person to their right. If you are one of these students and the person to your right nods to you:

- If you are sitting down, stand up.
- If you are standing up, turn and nod to the person next to you, and sit down.

If you follow these rules carefully you can count through every binary number from 0 to 255.

Notes for the teacher

Make sure you count slowly. The students will do their best to keep up.

Variations on this activity

Instead of using standing and sitting to represent 1 and 0 you could hold up cards with 1s and 0s on them or you could use flashlights that you turn on and off. The boys in this picture are turning around to display a 0 on their back or a 1 on their front.
Convert binary to denary

Introduction

You have learned about binary and denary numbers. In this section you will learn how to turn binary numbers into denary form.

Position values

You have seen that the value of a bit in a binary number depends on the position of the bit in the number. This table shows the position value of the eight bits that make up a byte.

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To convert a binary number into denary, you write the number into a table like this one. Make sure the number ends in the column on the right. Any empty columns should be at the left of the table. Put a 0 in any empty column.

Here is an example: the binary number 11010.

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

To turn this number into a denary number:

- find all the columns that have a 1 in them
- add the column values together.

The columns are 16, 8 and 2:

\[16 + 8 + 2 = 26\]

This shows that the binary number 11010 is the denary number 26.

16-bit numbers

The example on this page shows a number that uses up to eight bits (a single byte). One byte of data can hold any number from 0 to 255. Of course a computer can process numbers that are much larger than this. To do this the computer links several bytes together to make a large enough storage space.

You should learn how to convert numbers as large as 16 bits (two bytes). Remember that as you move to the left of a number, each bit has a value twice as big as the one before it.

Here are the position values of the bits in a 16-bit number.

<table>
<thead>
<tr>
<th>32 768</th>
<th>16 384</th>
<th>8192</th>
<th>4096</th>
<th>2048</th>
<th>1024</th>
<th>512</th>
<th>256</th>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You do not have to remember all these values. If you ever need to find the value of a 16-bit number you can work out the position values by starting at 1 in the right-hand column, and doubling the value as you count to the left (using a calculator).
To convert a 16-bit number into denary, create this table. Enter the 16-bit number into the table. Then add together the values of any column with a 1 in it.

**Test yourself**

1. Convert these 8-bit binary numbers into denary:
   - 0 0 0 1 1 0 0 1
   - 0 1 0 1 0 0 0
   - 1 0 0 1 0 0 1 1
   - 1 1 1 1 1 1 1 1

2. Here is the biggest number you can make with 16 bits. Convert this number to denary:
   - 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

**Learning activity**

Write a handout or a presentation to teach young learners how to convert 8-bit binary numbers into denary.
Convert denary to binary

Introductions
You have learned to convert binary numbers into denary. In this section you will learn how to turn denary numbers into binary.

Converting denary to binary
There is more than one way to convert denary numbers into binary. You will learn a method that uses the position values table.

Start with the denary number you want to convert. For example, let us convert the number 40 to binary. Next look at the table of position values. What is the largest value you can subtract from 40 (without making a minus number)?

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The largest value you can subtract is 32, so you write a 1 in the “32” column.

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Now take away the value. In this example 40 – 32 = 8.
What is the next largest value you can take away? It is 8, so you write a 1 in the “8” column.

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8 – 8 = 0, so the conversion is complete. Write a 0 in every other column.

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

This shows that 40 in denary is 00101000 in binary.

Simplify the process
To simplify the process, complete all the subtractions and then transfer the values to the position table.

For example, let us convert the denary number 99 into binary:

\[
\begin{align*}
99 &- 64 = 35 \\
35 &- 32 = 3 \\
3 &- 2 = 1 \\
1 &- 1 = 0
\end{align*}
\]
Now mark the numbers you have taken away in the position table. Write a 0 in every other column.

<table>
<thead>
<tr>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

This shows that 99 in denary is 01100011 in binary.

**Summary of the method**

To convert a denary number to binary, you must remember the column values of the binary table. Start with the largest value you can subtract from the number without going below zero.

Take the remainder from this subtraction. Continue to subtract numbers and take the remainder, until you have got to 0.

Look at all the numbers you have subtracted. Put a 1 in the matching columns of the binary table. Put a 0 in all the other columns.

**Test yourself**

1. Convert these denary numbers into 8-bit binary numbers:
   - 31
   - 55
   - 70
   - 101

2. Convert 500 into 16-bit binary.

**Learning activity**

Make a class handout to tell young learners how to convert denary numbers into binary.