BINARY & HEXADECIMAL
WORKBOOK

compsciworkbooks gcse

COPYRIGHT NOTICE

HOW TO USE THIS BOOK

1.

Open this workbook.

2.

Read the explanations, try and understand what's going on.

3.

Attempt the questions.

4.

Scan the QR code next to each question to see a video of the correct way of answering the question.

5.

Mark your answers and make corrections.

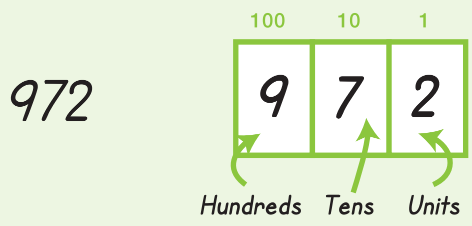
6.

Impress everyone with your amazing level of knowledge.

NUMBER SYSTEMS

DENARY

We humans we use a denary number system. This means that each bit in the number system can hold ten different values. Think back all the way to primary school when you were first learning to do simple sums and we'll see if we can make some sense of it.

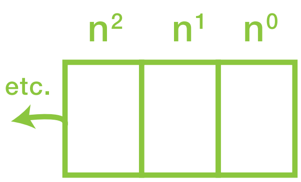


Possible values in one bit

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

So the number 972 is 9 Hundreds, 7 Tens and 2 Units. This is sensible right? I mean it's only the way you've been counting since you were a child so it makes sense.

Where do the place values come from then? Well it's simple mathematics:



Where n is the number of possible values in one bit, this means n multiplied by itself a number of times. Notice how the power of n increases the further to the left we go. In denary n=10 so those powers look like this:

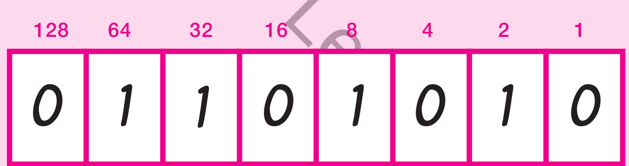
$10^2, 10^1, 10^0$ which is the same as $100, 10, 1$

Because of this we call denary a base-10 number system. Denary is sometimes refered to as the decimal number system.

BINARY

Computers count a little bit differently to humans, they have switches as their smallest parts and this means that they only allow the numbers 0 and 1 (off and on) in each bit. With only two numbers to chose from we can work out the place values of this base-2 number system.

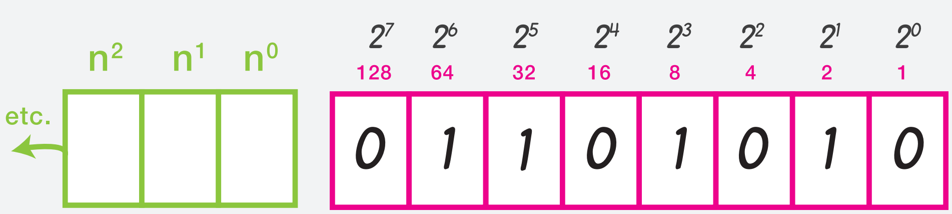
Because binary only uses two values in each bit, the numbers can be quite large, in most cases we group binary numbers into groups of four (called nibbles) or eight (bytes). The standard exam question will expect you to use an 8-bit byte, this has 8 bits.



Possible values in one bit

0, 1

What this example means is that there are zero 128s, one 64, one 32, zero 16s, one 8, zero 4s, one 2 and zero 1s. We can think of it like this - when a bit contains a 1 it is turned on and the place value of all the bits that are turned on are added together to give us our denary equivalent. Interestingly the place value comes from exactly the same calculation as in denary, only now the value of n is 2 because each bit has just two possible values.



This makes it really easy to remember the place value, because if we start from the smallest value on the right hand side, every time we move one to the left the place value doubles. It means you don't actually have to remember the entire sequence!

BINARY TO DENARY

Any question that requires that you convert from binary to denary will start by giving you an 8-bit binary value. For this example we will use the binary number 01101010.

01101010

The first job is to draw out the place value table for binary values.

You should then copy the binary number into the table.

128	64	32	16	8	4	2	1
0	1	1	0	1	0	1	0

You then need to write down a simple addition sum that uses the place values of all bits that are set to 1.

$$64 + 32 + 8 + 2$$

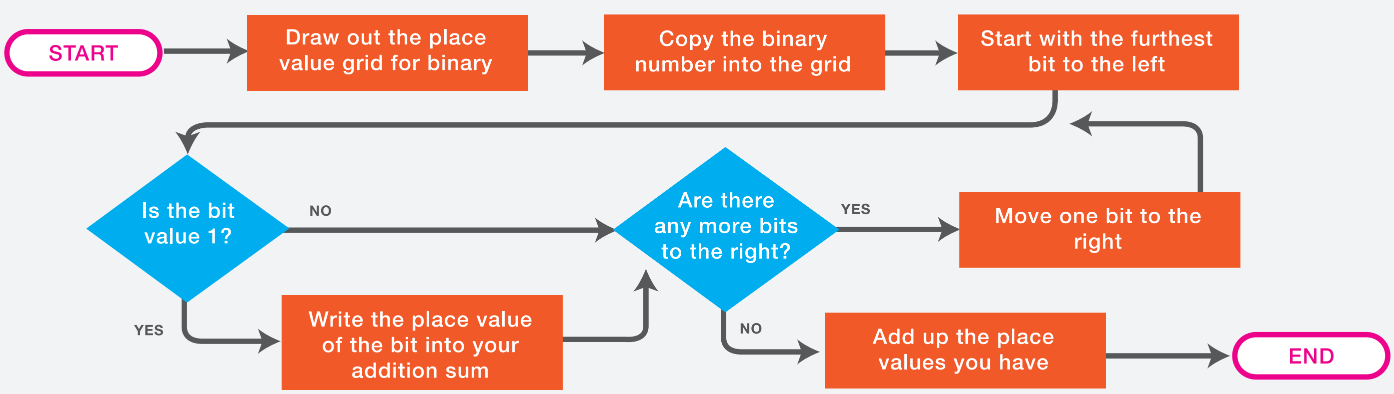
Then simply work out that sum to find your denary value.

$$64 + 32 + 8 + 2 = \underline{106}$$

This means that the binary value 01101010 can be written in denary as 106.

FLOWCHART

If you learn that process then you really cannot go wrong, to practice let's look at the technique in the form of a flowchart and then try it out on the worked example below.



WORKED EXAMPLE

Convert the 8-bit binary value 11001101 to denary.

128	64	32	16	8	4	2	1
1	1	0	0	1	1	0	1

$$128 + 64 + 8 + 4 + 1 = \underline{205}$$

QUESTIONS

1. Convert 1100 to denary.

2. Convert 1001 to denary.

3. Convert 00001111 to denary.

4. Convert 0000011 to denary.

5. Convert 00001110 to denary.

6. Convert 01011000 to denary.