Binary Worksheet

# Starting off

Computers store all of their information using REALLY tiny little switches. These switches can be either on, or off. Something that can only have 2 ‘states’ is said to be Binary.

Computers use Binary to represent all of the data that they store. We’re going to start off looking at binary numbers.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|  |  |  |  |  |  |  |  |

Lets look at the number 27. The way we normally write numbers is called DENARY. We can build a similar table to the one above for denary. You have probably seen one like it at Primary School:

|  |  |  |
| --- | --- | --- |
| Hundreds | Tens | Units |
| 0 | 2 | 7 |

A computer stores this number in memory using binary, it looks like this:

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

The ‘switches’ for number 16, 8, 2 and 1 are all turned ‘on’. 16+8+2+1 is 27. 00011011 is the number 27 in binary.

# Have a go

Convert the number 33 into binary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|  |  |  |  |  |  |  |  |

Convert the number 102 into binary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|  |  |  |  |  |  |  |  |

Convert the number 73 into binary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|  |  |  |  |  |  |  |  |

# Now try the other way

Convert these binary numbers into their denary equivalents:

|  |  |
| --- | --- |
| 10100101 |  |

|  |  |
| --- | --- |
| 00000101 |  |

|  |  |
| --- | --- |
| 01101101 |  |

|  |  |
| --- | --- |
| 11101100 |  |

# Question

All of these binary numbers have been 8-bit binary numbers, because there are 8 ‘slots’ to hold a 1 or a 0. What is the highest number you can represent with just 8 bits?