**Representing Data**

**Introduction**

[](http://www.google.co.uk/url?sa=i&rct=j&q=humans&source=images&cd=&cad=rja&docid=KsFmik4pqSGqQM&tbnid=mPaHW5wFdjz0lM:&ved=0CAUQjRw&url=http://endoftheamericandream.com/archives/the-green-agenda-is-about-getting-rid-of-as-many-humans-as-possible/the-green-agenda-is-about-getting-rid-of-as-many-humans-as-possible&ei=jfTXUaKjGMyY1AXB04DQDw&psig=AFQjCNFMjIjhdXh74oWhWN4QTlXfvQqfFg&ust=1373193715236953)

**Sight**

**Autonomic Nervous System**

**Memories**

**Expressions**

**Touch**

**Smell**

**Limbs**

**Speech**

**Hands**

**Outputss**

**Inputs**

**Brain**

**Humans**

[](http://www.google.co.uk/url?sa=i&rct=j&q=matrix&source=images&cd=&cad=rja&docid=-vkunPTK4RbAuM&tbnid=O8EsK1rC5KmogM:&ved=0CAUQjRw&url=http://www.taodirectory.co.uk/cracking-the-matrix.html&ei=P-rXUY3tApDK0AWbr4CABw&psig=AFQjCNHm3WhBWncTgwkzAkqYLbpeqDULDQ&ust=1373191054893530)

**Mouse**

**RAM**

**ROM**

**Flash Drive**

**Hard Drive**

**Sensor**

**Keyboard**

**Control System**

**Robot**

**Printer**

**Monitor**

**Outputss**

**Inputs**

**CPU**

**Computers**

Just like humans, computers perform tasks based on what information is presented to them. A human gathers information using its sight, smell, touch, taste and hearing sensors. The brain processes this information and outputs it in the way of speech, expression and movement. The brain has two memory types which it uses for storing information.

The first type which you will be familiar with are your memories. These are constantly changing and being updated as you live your life. These are the memories which you draw upon to make decisions and the memories that you use to learn. This is your Random Access Memory (RAM). F or longer term memories you store in your brains equivalent of a hard drive.

The second type of memory is less familiar and does not update or change. This can be thought of as your operating system. This type of memory controls all of your body’s subconscious functions such as your breathing and heart rate. Without this in built operating system you would just forget to breath when you fell asleep! This memory can be thought of as your Read Only Memory (ROM).

RAM is short term memory that helps the CPU with Processing

ROM cannot be changes and is used to contain the commands that keep the computer operating.

**Data Information Knowledge**

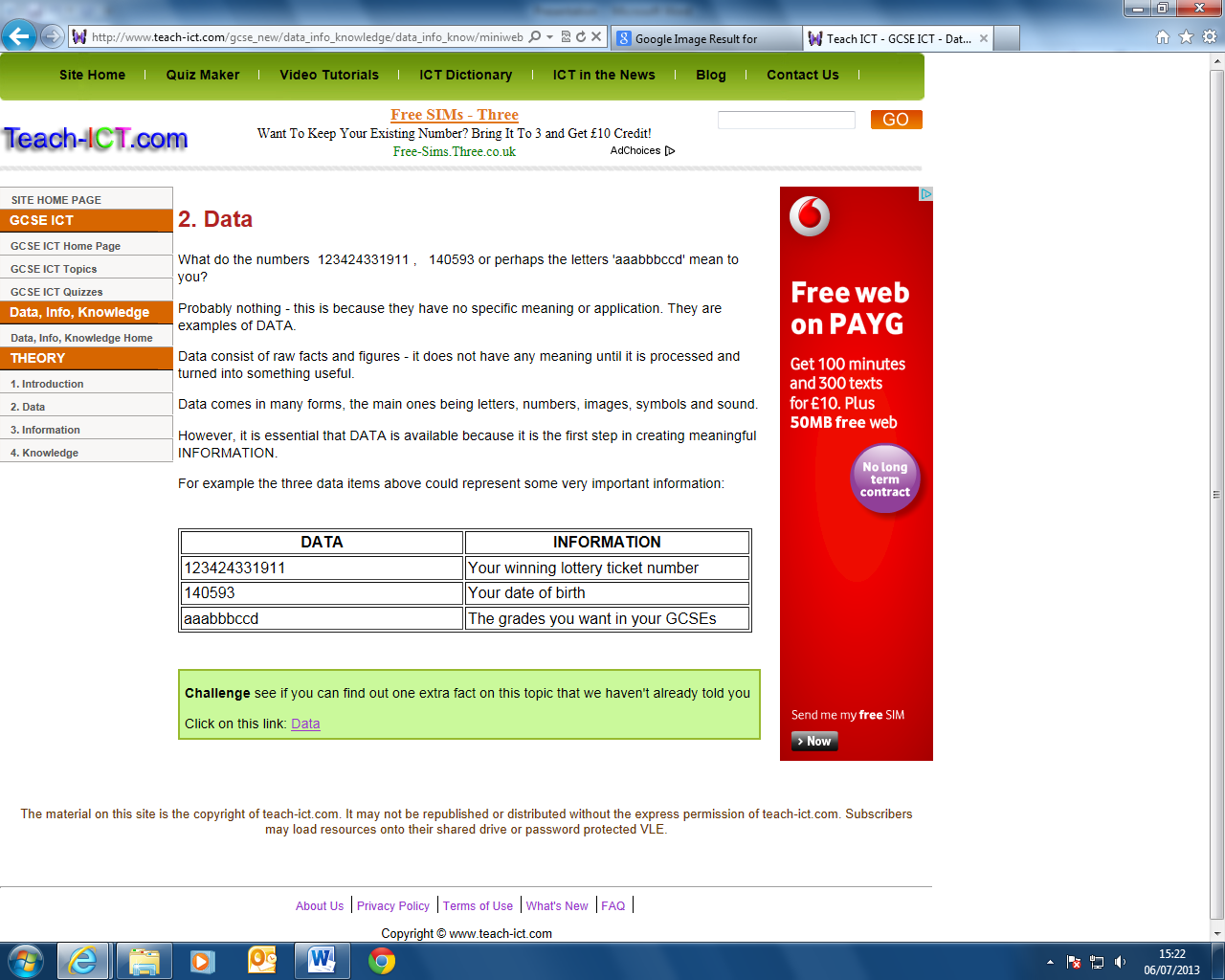
**Data**

Data is/are the facts of the World. For example, take yourself. You may be 5ft tall, have brown hair and blue eyes. All of this is “data”. You have brown hair whether this is written down somewhere or not.

In many ways, data can be thought of as a description of the World. We can perceive this data with our senses, and then the brain can process this.

Human beings have used data as long as we’ve existed to form knowledge of the world.

Until we started using information, all we could use was data directly. If you wanted to know how tall I was, you would have to come and look at me. Our knowledge was limited by our direct experiences.



**Information**

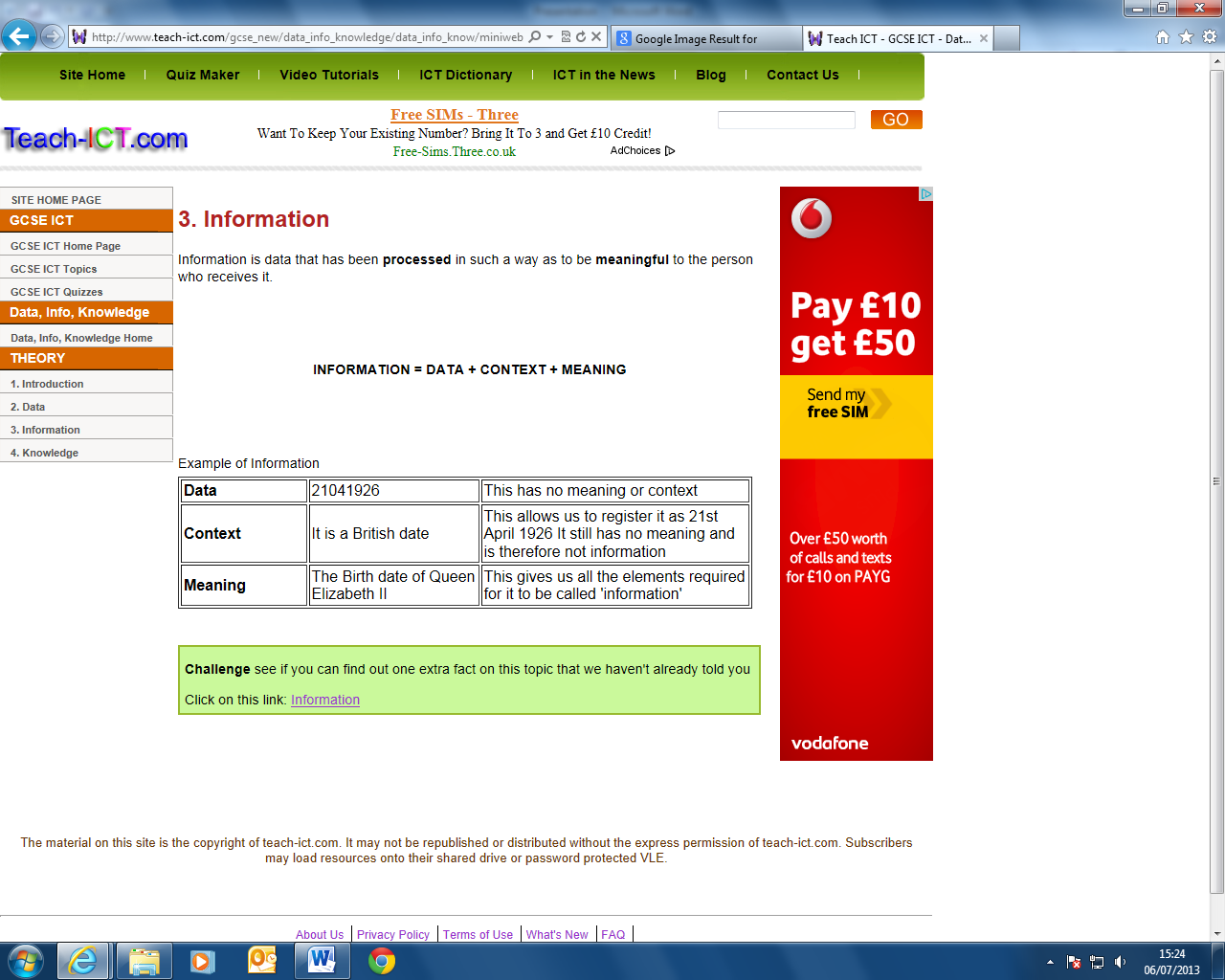
Information allows us to expand our knowledge beyond the range of our senses. We can capture data in information, then move it about so that other people can access it at different times.

Here is a simple analogy for you.

If I take a picture of you, the photograph is information. But what you look like is data.

I can move the photo of you around, send it to other people via e-mail etc. However, I’m not actually moving *you* around – or *what you look like*. I’m simply allowing other people who can’t directly see you from where they are to know what you look like. If I lose or destroy the photo, this doesn’t change how you look.

So, in the case of the lost tax records, the CDs were information. The information was lost, but the data wasn’t. Mrs Jones still lives at 14 Whitewater road, and she was still born on 15th August 1971.



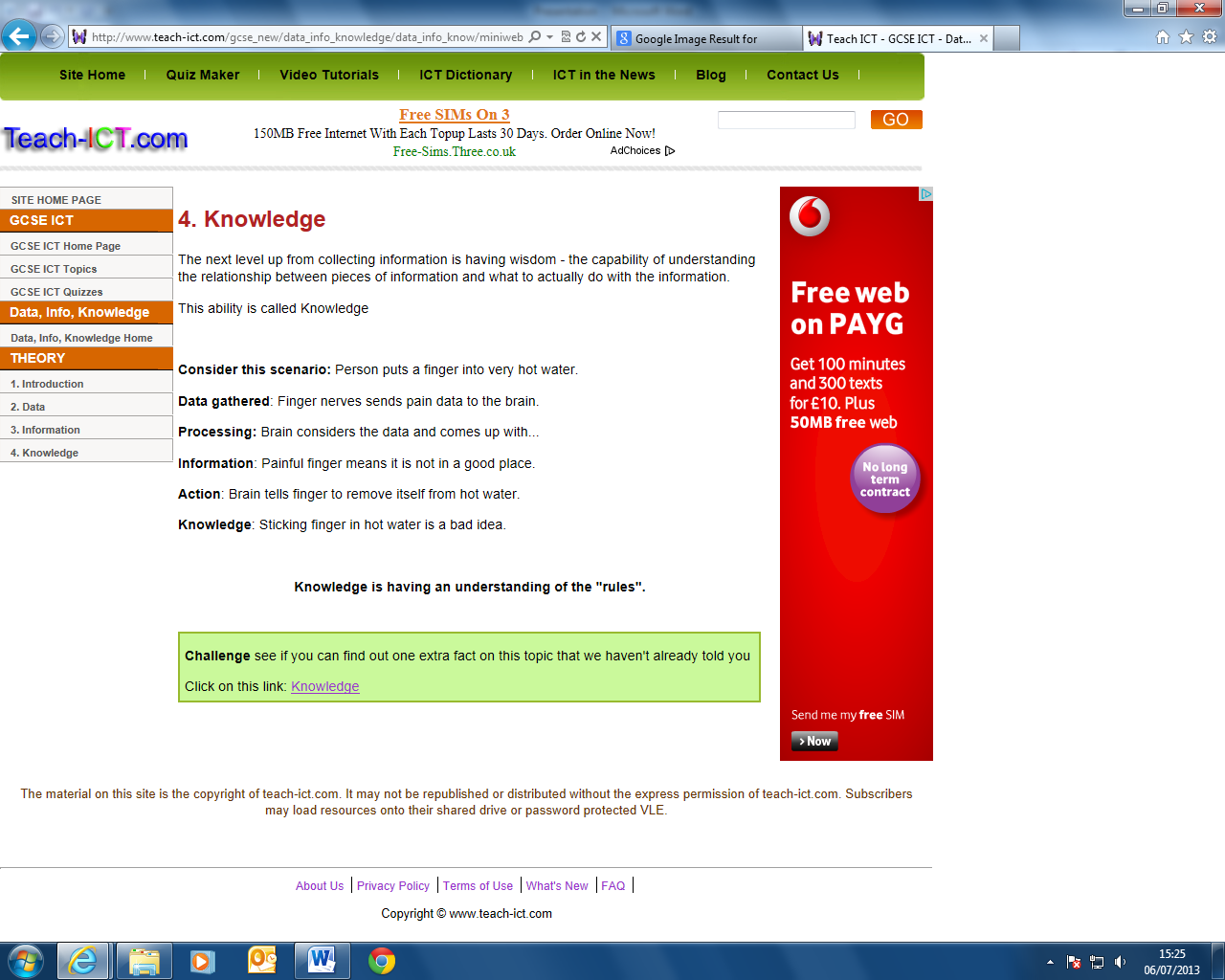
**Knowledge**

Firstly, let’s look at Knowledge. Knowledge is what we know. Think of this as the map of the World we build inside our brains. Like a physical map, it helps us know *where* things are – but it contains more than that. It also contains our beliefs and expectations. “If I do this, I will probably get that.” Crucially, the brain links all these things together into a giant network of ideas, memories, predictions, beliefs, etc.

It is from this “map” that we base our decisions, not the real world itself. Our brains constantly update this map from the signals coming through our eyes, ears, nose, mouth and skin.

You can’t currently store knowledge in anything other than a brain, because a brain connects it all together. Everything is inter-connected in the brain. Computers are not artificial brains. They don’t understand what they are processing, and can’t make independent decisions based upon what you tell them.

There are two sources that the brain uses to build this knowledge - information and data.



**Why does it matter that people mix them up?**

When people confuse data with information, they can make critical mistakes. Data is always correct (I can’t be 29 years old and 62 years old at the same time) but information can be wrong (there could be two files on me, one saying I was born in 1981, and one saying I was born in 1948).

Information captures data at a single point. The data changes over time. The mistake people make is thinking that the information they are looking at is always an accurate reflection of the data.

By understanding the differences between these, you can better understand how to make better decisions based on the accurate facts.

**In Brief**

**Data:** Facts, a description of the World  
**Information:** Captured Data and Knowledge  
**Knowledge:** Our personal map/model of the World

**Software**

We have looked briefly at Hardware but equipment such as this needs instructions for it to do something – Programs or software. In ROM is the smallest and first piece of software – Bootstrap loader. This program is set to run (execute) when the computer is first turned on. This program tells the computer to load its operating system.

**Software**

Sometimes abbreviated as S/W, software is a collection of instructions that enable a user to interact with the computer or have the computer perform specific tasks for them. Without software, the computer would be useless. For example, without your Internet browser software you would be unable to browse the Internet and without a software operating system the browser would not be able to run on your computer.

**Bootstrap loader**

Alternatively referred to as bootstrapping, boot loader, or boot program, a bootstrap loader is a program that resides in the computers EPROM, ROM, or other non-volatile memory that automatically executed by the processor when turning on the computer. The bootstrap loader reads the hard drives boot sector to continue the process of loading the computers operating system. The term boostrap comes from the old phrase "Pull yourself up by your bootstraps."

The boot loader has been replaced in computers that have an Extensible Firmware Interface (EFI). The boot loader is now part of the EFI BIOS.

**Operating system ABCs**

An operating system or OS is a software program that enables the computer hardware to communicate and operate with the computer software. Without a computer operating system, a computer and software programs would be useless

**Binary**

Computers are two state devices they contain millions of switches that are either ON or OFF. As there is only two states the only information computers can only be recognise the terms 1 and 0. Where we use 10 different symbols to represent our base 10 number system computers have two states, 1 representing on and 0 representing off. Therefore computers work in Base 2 number system called binary.

Computers only understand instructions and data in binary and the first computers had to be programmed using binary instructions and the data that was to be processed had also to be in binary. This was very slow and difficult and errors in programs often meant rewriting large portions of code.

To speed up the process, low level programs (often called assembler languages) were developed that made the writing of instructions much easier and quicker. Every binary coded instruction is replaced by a short English word.

*The definition of a low level language is that one, and only one, assembler instruction replaces a machine language instruction (binary).*

Assembly language programming is beyond the scope of current GCSE courses but an understanding of binary is needed.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9  Decimal / Denary (Base 10 number system) |

Binary (Base 2 number system)

|  |  |
| --- | --- |
| 0 | 1 |

Each binary unit is called a BIT. A BIT represents one binary unit (either a 1 or 0). Four BIT’s are called a NIBBLE and eight BIT’s are called a BYTE.

In a BYTE the 8 bits represent the most to least significant BIT.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |

Most Significant BIT Least Significant BIT

**Counting In Binary**

**Denery / Decimal**

Counting in the decimal system involves using units, tens, hundreds, thousands, millions etc. So to represent five million, two hundred thousand, three hundred and twenty seven we can break it down into the following components.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Millions | Thousands | Hundreds | Tens | Units |
| 5 | 200 | 3 | 2 | 7 |

Similarly thirty would be represented by the following components.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Millions | Thousands | Hundreds | Tens | Units |
| 0 | 0 | 0 | 3 | 0 |

**Binary**

As a computer can only recognise the states 1 or 0 representing numbers like this is completely useless. Binary works by assigning each increasingly significant BIT in the BYTE a value. The values are determined thus.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |

Most Significant BIT Least Significant BIT

The largest number a BYTE can represent is 255 (128+64+32+16+8+4+2+1)

**Representing numbers 0 to 255 in binary.**

**0 in binary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

0 = 00000000

**1 in binary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

1 = 00000001

**16 in binary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

16 = 00010000

**37 in binary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

37 = 32+4+1 = 00100101

**173 in binary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |

173 = 128+32+8+4+1 = 10101101

**255 in binary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

255 = 128+64+32+16+8+4+2+1 = 111111111

**Representing larger numbers in Binary**

Numbers larger than 255 can be represented by simply adding another BYTE. The largest number that can be represented by two BYTES is 65535.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 215 | 214 | 213 | 212 | 211 | 210 | 29 | 28 |  | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| 32768 | 16384 | 8192 | 4096 | 2048 | 1024 | 512 | 256 |  | 128 | 64 | 32 | 16 | 8 | 4 | 2 | Units |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |  | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |

The binary in this example 1010110110101101 = 32768+8192+2048+1024+256+128+32+8+4+1 = 44461

The larger the number the more BYTES need to be used.

**Representing Data**

Data can be categorised as numeric, alphanumeric characters or strings of characters (strings). Whatever the data, it will be represented by the computer in binary.

**How does a computer handle text?**

The text letters and symbols from a computer keyboard are assigned their own binary code.

A = 010000012 = 6510

B = 010000102 = 6610

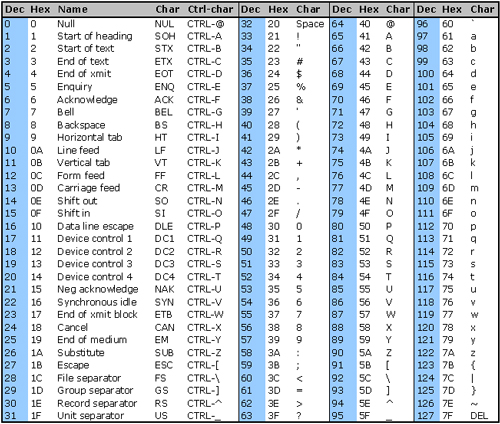
C = 010000112 = 6710

ETC………………………………………….

This is a standard code used by all computer manufacturers and is called ‘American Standard Code Interchange Information’ (ASCII). This code made the compatibility between different manufacturers hardware possible.

## Representing Text

To make the passing of data from one computer to another easy, a standard form of representing characters was adopted by computer manufacturers. The American Code for Information Interchange (ASCII).



**Hexadecimal**

As you can imagine inputting binary code into a computer would be a very difficult and time consuming exercise because to us reams and reams of 1’s and 0’s don’t make any sense. Your computer doesn't have a problem with it since it is all that a computer can understand. To make coding easier for your standard human programmer, binary is converted to Hexadecimal, making programs much easier to code.  
  
HEX converts to binary. Groupings of 4 binary places have a HEX value. If you've ever seen something like this:  
  
F3C2 in HEX, It would look like this in binary:  
  
1111 0011 1100 0010  
  
Here is a conversion chart of decimal, hexadecimal and binary:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Decimal** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| **Binary** | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| **Hexadecimal** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

That still doesn't answer how a computer uses HEX code.   
  
Computers are comprised of chips, registers, transistors, resistors, processors, traces, and all kinds of things. To get the binary bits from one place to the next, software programmers convert binary to hex and move hex values around. In reality, the computer is still shoving 1's and 0's along the traces to the chips.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Binary | 1011 | 0110 | 1110 | 1110 |
| Hexadecimal | B | 6 | E | E |

Along comes Assembly Language. In Assembly Language, you would move a HEX value into a register like so:  
  
Mov AX,03  
  
The command is MOV or move.  
  
The item to move is the value of 3  
  
Where? Into the AX register.

**Higher Level Languages**

The computer uses an interpreter to convert the higher level language that is easier for us to understand in machine code that the computer can understand. In the same way that the keys on the keyboard are assigned a binary code the commands in the assembler language are assigned binary codes.

[](http://www.google.co.uk/url?sa=i&source=images&cd=&cad=rja&docid=xZs5vP_BXt5cOM&tbnid=n6-1YAWjyCzmQM:&ved=0CAgQjRwwAA&url=http://www.ehow.com/how_7413463_learn-binary-code.html&ei=4CLYUazyJcbw0gXunIHoCw&psig=AFQjCNFvFR5hwOom8y2okEf2qJ3kTmYtHA&ust=1373205600669648)

High Level Language

Assembler Language

Machine Code

Boot Strap Loader

CPU

For a set of instructions (program) the interpreter translates on the fly. The entire program (source code) is translated into machine code then the machine code version (object code) is executed.

Compiler

Source Code

Source Code