What is an Object Anyway?

In this first lecture we are going to think about these things called objects. We are going to be using objects as a key component of our programming. What are they and how do they fit into programming?

Let's start though with a different question.

What is a word?

Take a single word and think about that word.

Think about the following word.

shop

What springs to mind?

We might think about a specific shop.

Or we might think about the act of shopping...

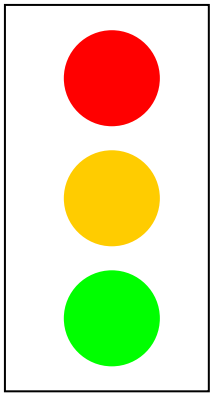
A word on its own isn't much use to us however when we combine it with other words we can create instructions.

"I want you to go to the shop and get me a tin of peas."

The same is true of symbols and signs.

What is this?

What if we place the same shape in a context? Now what is it?



What in this context does a red circle mean? It means stop.

Traffic lights and words on a page are all the same they are all symbols that we look at and assign meaning.

In this first lecture we will look at why we need programming languages and relate them to the business of processing data within a computer system.

What is Programming?

Programming is about using words and symbols to communicate with a Computer

As with our example above we use words / symbols to create instructions or functions that the computer will follow and perform some task.

And now a lesson from Harry Potter...

What makes Hermione Granger so annoying?

The problem is that in Harry Potter to perform a spell you must say the words perfectly.

She invariably did, much to the annoyance of her peers.

In Harry Potter the magic is invoked by saying the right words correctly.

The same is true of programming we need to state the right words perfectly.

This is going to be your first challenge in learning to program.

As a beginner, you may think that you are saying the right words but when it comes to it the outcome will not be what you expect, this is perfectly normal.

Most of us end up feeling more like this Ron Weasley!

Computer Programming introduces a new Vocabulary

Consider the following words...

> "Shop, street, road, post box, lamp post, post code, house, flat, house number, left, right ahead, distance, time, street sign, landmark."

These words make up the vocabulary of the streets and roads. We are familiar with these words and can easily make up instructions (functions) using these words.

> "You want to walk down that street for 10 meters then turn left at the post box. You will then see a sign to the railway station. Follow that sign for 5 meters and the shop is on the left."

Consider the following words...

> "Class, object, assignment, sequence, variable, RAM, parameter, data type, function, method, property, selection, sequence, integer, string."

These are some of the words used to describe the world of the computer program.

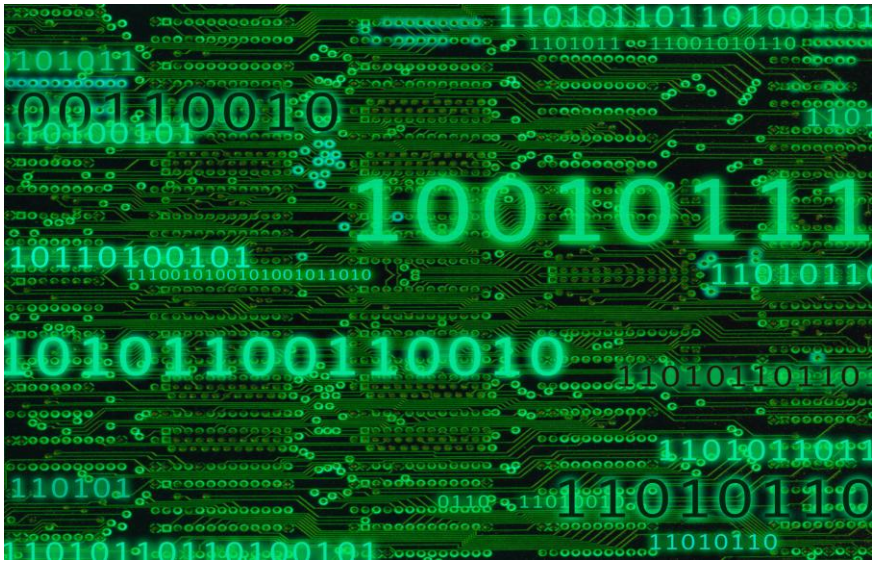
If we use these words and concepts in the correct way and give them to a computer to process we get a computer program that does something.

Programming is all about Data

Programming is about using the right words to control the data within the computer system.

Binary

At its lowest level data is stored and represented as zeros and ones called binary.



As a numbering system, we only have two digits to play with, 0 and 1.

0	=	0
1	=	1
10	=	2
11	=	3
100	=	4

By combining these binary digits or BITS in different ways we get to represent other things from simple mathematical calculations...

1 + 1 = 2

to complex multimedia systems for example BBC iPlayer.

These BITS are used in such a way that not only do they control what we see in terms of text, colour, and audio/video content they also determine such things as the security and navigation around the system.

To make the BITS behave in such a way that we see the content above we need to control them by means of a programming language.

The right words organised in the right way.

We take the noughts and ones – the data – and transform them into something meaningful – the information.

The thing about the data though is that the information is determined by the type of hardware using them.

We might see a sequence of zeros and ones like so...

01001010100101111010100100101001001001010101011111110

Out of context they don't mean much to us however if sent to the right hardware the sequence could be part of the following...

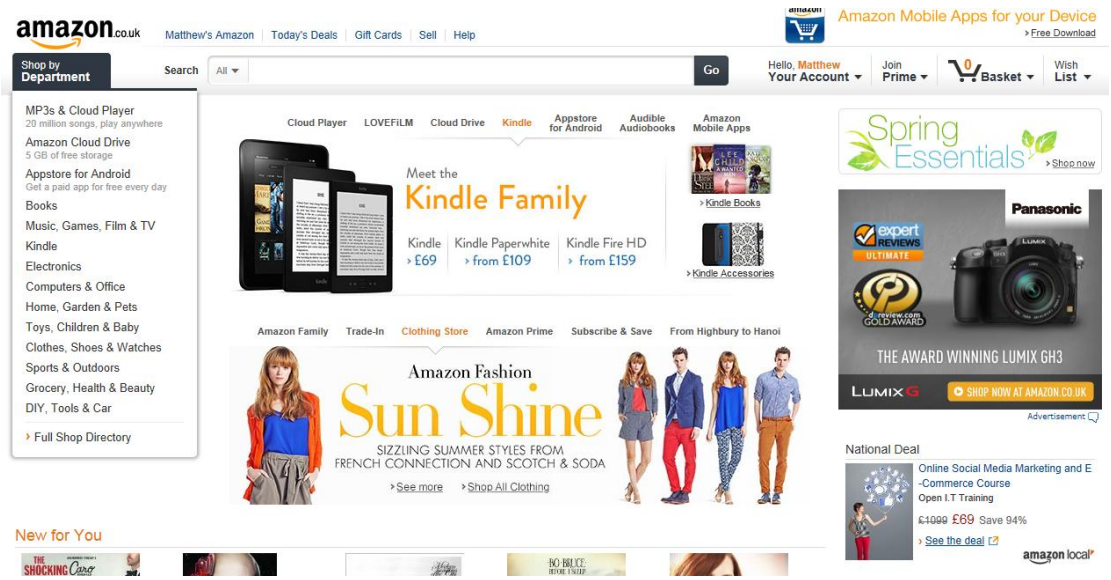
- A mathematical calculation
- Part of an MP3
- A section of video
- Some text

It all depends on what hardware is making use of the data.

Users Don't Care About the Data

As users of computer systems, we don't tend to think about the zeros and ones at all.

When we go online to but some goods we don't really think about the data that makes it all work, we simply want to get on with a job at hand.



As a user of a system we are not interested in all of the noughts and ones flying around under the bonnet. We look at a site like Amazon and what we really want to do is make use of its services.

The Computer Programmer

The thing with this data is that it doesn't simply organise itself. There is a person who has created functions to tell the data how to behave so that they make some sort of useful application.

As new students on this module you will be taking your first steps as programmers learning how write functions the computer understands.

The problem comes though that as a programmer you will start to be exposed to these zeros and ones making up the data in a way that as a system user you were oblivious.

Human beings do not think naturally in binary!

Consider the not very funny joke...

"There are 10 kinds of people in the world, those who understand binary and those who do not."

The point is that we as human beings tend to think in base ten not base two. If we want to count to ten it's easy – 1,2,3,4,5,6,7,8,9,10. Doing the same in binary comes less naturally.

So, considering that we as programmers are going to have to start thinking about this binary world and we as human beings are not naturally going to find this easy what do we do?

Well, this is where programming languages come in.

A modern programming language will allow us to control the data within the hardware of a computer in such a way that it is "relatively" painless for us.

Programming languages provide us with a set of commands that allow us to control the data and hardware in something that is reasonably close to normal language.

How to shut down a Computer

Let's consider a very simple task that we want a computer to perform. The task is so simple that we probably don't ever think much about it.

I want the computer to shut down.

Now before we start thinking about any computer program that might do this what I want to start with is how we as human beings would handle this task.

If I wanted to turn a computer off I would simply get up walk over to the computer and press the power button (or select start and shut down).

Ok, now what if I wanted to instruct one of you to turn off the computer?

I could say "stand up walk over to the computer and press the button on the front once."

This is a very simple set of instructions that we could all follow and is expressed in natural language, i.e. the language that we as human beings use to communicate.

Natural Language and Computers

The big problem comes if we wanted to send that set of instructions to the computer to tell the computer to turn off.

We could stand next to the computer and say, "ok computer turn your self off now".

And we would find that not very much would happen. We also have several other problems to consider not least of which being what do the words actually mean?

Does the computer understand the words we have used?

Computer
Turn
Off
Now

The problem is that natural language is quite sophisticated.

As human beings, we tolerate a great deal of imprecision in the use of language.

Consider the following

Off computer turn now
OK puter turn thesel ov now
Cmptr trn yrslf ff nw

All the above statements are easy for us to understand as humans but a computer would have no idea.

We could install speech recognition software onto the computer but that would still fail given for example a strong regional accent.

A big part of the problem here is that computers are really quite thick.

They can do some amazing things but when it comes to understanding what we as human beings would like them to do they are quite challenged on the communications front.

So, if we cannot use natural language to tell a computer what we want it to do we have to have another approach.

Symbols and Meaning

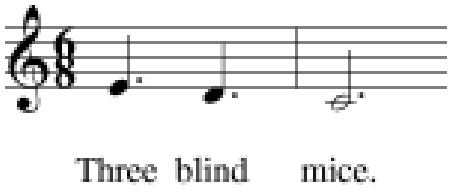
We see the word “shop” and we know what a shop is.

We see a red traffic light and we know we need to stop.

We use symbols to refer to real things all the time in our daily life.

In music, we use notation to represent the structure of a piece of music.

The musical notation below...



“Represents” the tune “three blind mice”

The notation is NOT the tune! It only becomes the tune when the data is processed through a musical instrument or voice.

A musician can understand the meaning of the symbols and follow the instructions to reproduce the tune.

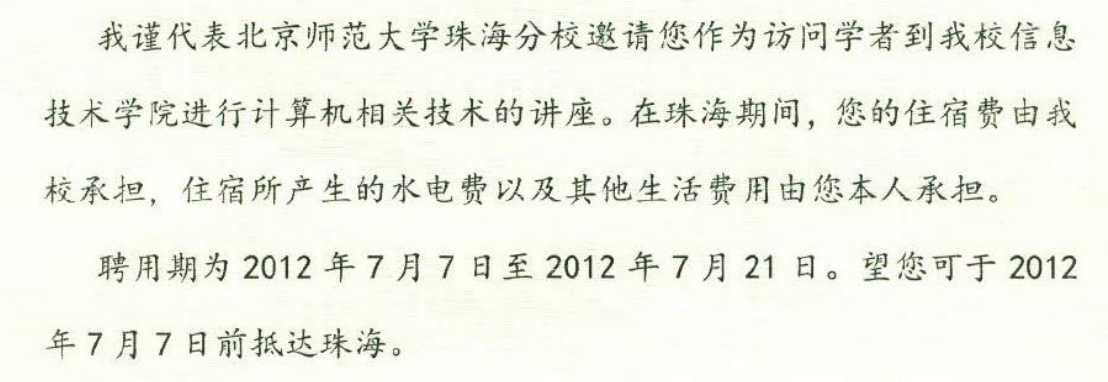
Consider a postal address.

De Montfort University
The Gateway
Leicester
LE1 9BH
0116 255 1551

This address is not the location it is a set of words (symbols) that represent the actual location.

It is the meaning of the symbols that represent a physical location.

The problem comes if we do not understand the symbols used...


我谨代表北京师范大学珠海分校邀请您作为访问学者到我校信息技术学院进行计算机相关技术的讲座。在珠海期间，您的住宿费由我校承担，住宿所产生的水电费以及其他生活费用由您本人承担。

聘用期为 2012 年 7 月 7 日至 2012 年 7 月 21 日。望您可于 2012 年 7 月 7 日前抵达珠海。

If we do not understand the symbols, the meaning fails to be communicated.

Initially sections of code will all look quite similar at first until you learn to understand the difference.

For example, can you spot the difference in the following code segments? What is their meaning?

Int32 Name;	-	Variable
Name();	-	Function
“Name”	-	A string of letters
.Name	-	A method or a property (Probably a property)
//Name	-	A comment
(Name)	-	A parameter

Syntax – Getting the Punctuation and Spelling Right

The computer is really, fussy and gets upset at the slightest error.

Look at the following simple section of code…

```
Int32 MyAge;
MyAge = 23;
```

```
if (MyAge > 21)
{
    lblError.text = "You are old enough";
}
```

If we try to run this program it will not work, can you spot why?

Below is the corrected version of the code…

```
Int32 MyAge;
MyAge = 23;
if (MyAge > 21)
{
    lblError.Text = "You are old enough";
}
```

Can you spot the fix?

The problem was that we used .text rather than, .Text when referencing the control lblError. In the case of Visual C# the language is case sensitive.

A tiny little typo in a program will almost always stop the entire program from working as the computer is too stupid to work it out.

One thing that is going to take a while to get used to is the fussy syntax of computer languages.

If we say something to a human being and get it slightly wrong the other person will probably get the idea of what we are trying to say.

The same is not true of computer systems. If we get one thing wrong in our code the computer will not be able to work out what we mean.

For example, in the section of code below some lines have semi colons at the end others don't ...

```
Int32 MyAge;
MyAge = 23;
if (MyAge > 21)
{
    lblError.Text = "You are old enough";
}
```

Knowing when to put semi colons in and when to leave them out will be quite a pain until you get the idea of what the rules are, the syntax.

Computer Programming

The above discussion raises several issues that directly relate to computer programming.

We want to create:

- A system that may be used to write a sequence of words allowing us to tell the computer what to do with the data
- A set of instructions that are simple and unambiguous as the computer is quite stupid
- Something that is reasonably easy for us as human beings to write and understand (we don't want the symbols to get in the way of the meaning)

Functions, Classes, Objects, Methods, Properties and Instantiation

Another problem with learning computer programming is that there is a whole new vocabulary to get to grips with.

Let's return to the problem of how we tell a computer how to turn itself off.

If we are to tell the computer to turn off then we need to decide on a way of thinking about the system so that we can use words to create functions.

Like with a street of houses we need a vocabulary to describe parts of the computer system.

There are a set of terms that need to be understood that allow us to describe the various parts of the computer system. Some of these terms relate to things we can easily understand, RAM, Hard Drive, Screen or Mouse. Some of these units are harder to grasp and relate to how we control and organise the data in the system.

Expanding your Programming Vocabulary

To start with we will introduce some important words that we will re-visit again and again.

- Functions
- Methods
- Properties
- Classes
- Object
- Instantiation

Functions

"A function is a named section of code that does one thing well."

Here is a function that multiplies two numbers together and returns the answer...

```
decimal Multiply(decimal FirstNumber, decimal SecondNumber)
{
    ///this function accepts two decimal numbers multiplies them together
    ///and returns the answer

    //declare a variable to store the answer
    decimal Answer;
    //perform the calculation
    Answer = FirstNumber * SecondNumber;
    //return the answer
    return Answer;
}
```

The function is a sequence of words that mean something to the computer. When the computer reads these words, it understands them and acts on them.

Don't worry about the detail of what is going on at this stage. Just remember a function is a named section of code that does one thing well, in this case to multiply two numbers together and return the answer called "Multiply".

Functions are the building blocks of a computer program. A single program is made of lots of functions working together to perform the task.

What do we put in a toaster?

Like bread we use functions to make other things.

For example

- Methods
- Properties
- Event handlers
- Test cases

Two varieties of function to consider at this stage are properties and methods.

Properties

A property allows us to get or change the settings of an object

For example, if we are working with data relating to students there may be a property called Count which tells us how many students we are dealing with.

```
//create an instance of the student class
clsStudents MyStudents = new clsStudents();
//variable to store the count of students
Int32 StudentCount;
//get the count of students
StudentCount = MyStudents.Count;
```

Methods

A method tells the object to perform an action

If we are dealing with data related to student records then we might have a method that allows us to add a new student to the system.

```
//create an instance of the student class
clsStudents MyStudents = new clsStudents();
//add a new student
MyStudents.AddStudent("Fred", "Bloggs", "P12345678");
```

Classes

A class is a file that contains related functions making them easier to manage.

Classes are like recipe books. We might have several recipe books for different parts of the world, Indian, Italian, Chinese etc. It would be unusual to have an Italian recipe in a book on Chinese cookery.

In the same way, we group our related functions into classes.

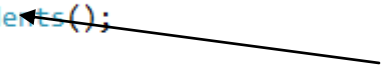
If we want to manage student data then we will create a set of functions along the following lines...

Methods
- .AddStudent
- .UpdateStudent
- .DeleteStudent

Properties
- .Count
- .HighestStudentGrade

What we would do is organise these functions into a single file called clsStudents. This is what we are referencing in the two examples above.

```
//create an instance of the student class
clsStudents MyStudents = new clsStudents();
//add a new student
MyStudents.AddStudent("Fred", "Bloggs", "P12345678");
```



So, what is an Object Anyway?

Understanding what an object is can be quite tricky at first.

One way to think about an object is that it is simply a word on the page.

```
//create an instance of the student class
clsStudents MyStudents = new clsStudents();
//add a new student
MyStudents.AddStudent("Fred", "Bloggs", "P12345678");
```

In the above section of code, we have created a word called MyStudents. This is an example of an object. It is just a set of letters / symbols forming a word that has meaning to the computer.

The word is something we decide upon as a programmer.

The thing about the word is that it has meaning to the computer.

In this case,What the word MyStudents is linked to the functions in the class file clsStudents.

This means that any instructions we write related to this word will trigger the functions in the class.

Instantiation

This process of creating an object from a class is called instantiation.

Coming back to the example of turning off a computer the code to do this might look like this.

```
//create an instance of our computer class called MyComputer
clsComputer MyComputer = new clsComputer();
//use the TurnOff method to turn the computer off
MyComputer.TurnOff();
```

The first line of code in green is called a comment and is written to help the programmer understand what is going on. When the program runs, the computer ignores comments.

The second line is the point where we make an instance of the class.

```
clsComputer MyComputer = new clsComputer();
```

The code that drives the process is in the class clsComputer.

We create a word MyComputer. When we refer to that word it makes the link between the word and the code in the class. When we invoke the functions of the object it triggers the code in the associated class.

The last line of code invokes the function/method “TurnOff” which in turn calls the code defined in the class and turns the computer off.

```
MyComputer.TurnOff();
```

Summary

We have covered a lot of different topics in this lecture and we shall return to these concepts as the module unfolds.

Don't worry about the detail of the discussion but take away with you some key words and over time their meaning should become clear.

Programming is the act of writing lists of words that control the data in the system.

We need a system for writing lists of instructions for the computer that makes reasonable sense to human beings.

We also have some key words for you to start getting to grips with…

Function	A function is a named section of code that does one thing well.
Property	Allows us to read or change the settings of the object
Method	Tells an object to perform an action
Class	A class file is a text file that stores functions for a specific task
Object	An object is the name we give to a class file so that we may access its methods and properties
Instantiation	The process of making an object from a class