CO452
Programming Concepts

Lecture 1

Introduction to Classes, Objects, Methods and Variables

Classes and objects

Classes are the **blueprint**

Unique objects can be created from this blueprint

Two parts of a class

Variables

Methods

Different terms!

variables / data / fields / attributes

methods / functions / behaviours / subroutines

Variables

What are variables?

Variables are storage areas for a single value

Values come in different formats – from whole numbers, decimal numbers, small numbers, large numbers, to words, letters and phrases

The value is stored at a memory address (RAM – temporary memory) which can then be accessed through the variable name

Whenever we refer to the variable name we refer to the value stored in memory

Variables can only store one value at a time

How do we define variables?

1. First of all, we have to define the type of data that we want to store – **data type**

2. Then we have to select a **meaningful name** which represents that data well

```
String  name;
```

1. Data types

int	whole numbers e.g. 10, 75, 200
double	decimal numbers e.g. 2.56, 0.314, 20.75
String	a class that represents more than one letter
char	single letter or number or symbol
boolean	true or false

2. Name

- Important to select names that describe the values they hold
- Best practice to start variable names with a letter or '_'
- Can’t have spaces in the variable name (use camel**C**ase or '_')
- Can’t be a reserved word (public, int, class)
- Beware of casing: ‘NAME’ and ‘name’ are different variables!

Encapsulation

public	• Can be referenced outside of a class
private	• Can only be referred to within the class
protected	• Can only be referred to with class hierarchy

Assigning values to variables

The assignment operator (=) is used to assign a value to a variable

The value on the right hand side of the '=' sign is ‘assigned’ to the storage area on the left hand side of the '=' sign

```
String name = "Nick";
int id = 21015672;
boolean isComplete = true;
```

Different to maths with trying to balance right and left hand sides of an equation…

Constants

Constants are initialised with a value, but then it’s not possible to overwrite that value later.

Setting values to be constant to prevent accidentally overwriting them.

In Java, use the keyword final and it’s also conventional to use UPPERCASE letters for an identifier.

```
public final int FEET_IN_MILES = 5280;
public final int HOURS_IN_DAY = 24;
public final int MAX_MARK = 100;
```

Methods

What are methods?

Blocks of code that represent one operation or behaviour

They may contain more than one line of code

This block of code is given a name – which when called - executes the statements within that block

It’s possible to pass data into methods as well as return data from a method

How to declare a method

```
/**
 * this is a comment that describes the method
 */
public void methodName ()
{
        //statements to go here
}
```

Passing data to a method

Declare a variable within the parentheses of a method to store data being passed in

That data is only available within the scope of that method

```
public void setName(String name)
{
        this.name = name;
}
```

Returning data from a method

Use the **return** statement within a method to pass data back to where this method was called from

Must **change the return type** of the method to match the type of data being returned:

```
public String getName()
{
        return this.name;
}
```

this

‘this’

‘this’ can be substituted for any given object created of the class

Rather than specifying one object that will be referred to every time the method is run, this can refer to the object that the method is being called on

```
public String getName()
{
        return this.name;
}
```

Constructor

What is a Constructor?

- ❖A constructor is a method which has the same name as the class
- ❖The constructor is called when we create an object of the class
- ❖The constructor does not have a return type

```
class House
{
        private int number;                          Variable

        public House(int number)                     Constructor
        {
                this.number = number;
        }

        public int getNumber()                       Getter method
        {
                return number;
        }
}
```

How to create an object of a class

Creating an object of a class

Now in code, we have to declare a **variable (object)** of a type (in this case, our class **House**)

```
House house = new House(5);
house.getNumber();

House mansion = new House(1);
mansion.getNumber();
```

Output

Output data to the console

We can use the print method to display data on the same line of the screen

```
public void print()
{
        System.out.print("House number ");
        System.out.print(number);
}
```

Concatenation and println

We can also use println to print text and then move the cursor to the next line

We can also join (concatenate) different types of data with the plus sign (+)

```
public void print()
{
        System.out.println("House number " + number);
}
```