**2 - Classes and objects**

* A code in object-oriented language can be changed without making changes to the tested code by adding new classes and methods.
* Java has inheritance, that is, a class is the Superclass which has Subclasses. This Subclass helps inherit methods from the Superclass, that is, if the super class has a functionality the sub classes inherit the same functionalities as well.

**Super Class**

**Sub Class**

**Sub Class**



**Inheritance**

* Method overriding: subclass shows different implementations from superclass in the properties inherited.
* Things that the object knows about itself is called instance variables.’
* Things that the object can do are called methods.
* Instance is another way of saying object
* A class is not an object but its used to construct them.
* A class is a blueprint of an object. It tells the virtual machine how to make an object of that particular type.
* We need 2 classes to use an object. One class for the type of object we want to use and another class to test your new class. The tester class is where we put the new method.

Below is an example code:

//Writing the class

Class Human {

String name;

Int age;

void talk ()

{

System.out.println(“hello”);}

}

//Writing the tester class

class HumanTest{

public static void main (String [] args)

Human pravallika= new Human ();

pravallika.age=22;

pravallika.speak();

}}

* The 2 uses of main are:

1. To test your real class
2. To launch/start your java application

* A real Java application is objects talking to objects.
* Each time an object is created in java it goes into an area of memory known as the heap.
* All objects no matter when, where or how they’re created live on the heap. Java heap is called the garbage collectible heap.

**3 Primitives and References**

* Variables are of 2 types

1. Primitive
2. Reference

* The type of the variable should always be declared. We cannot compile 2 variables of different data types.
* Variables must have a name.
* A variable is like a container, that is, it can contain some value.
* Byte=8
* Short=16
* Int=32
* Long=64
* Float=32
* Double = 64
* Anything that is declared as float should be with an ‘f’ as the suffix or java takes it as a double value.
* The following are rules for naming->

1. It must start with a letter, underscore, or dollar sign ($). You can’t start a name with a number.
2. After the first character we can use numbers.
3. It can be anything except one of java’s reserves words.
4. An object reference variable holds bits that represent a way to access an object.
5. It does not hold the object itself but holds the address of the object.

* An array is a collection of similar data types.
* Eg: nums= new int[7]

Num[0]=1; Index



Num[1]=2;

Num[2]=3;

Num[3]=4;

Num[4]=5;

The number in the bracket is the index of the array.

* Arrays are always objects
* A reference variable has a value of null when it is not referencing any object.

Example code:

String dogs [] = new dogs [3];

dogs [0] = ‘Hachi’;

dogs [1] =’Boo’;

dogs [2] =’casper’;

**4- Methods use instance variables**

* A class describes what an object knows and what an object does.
* A class is the blueprint of an object.
* Every instance of a class has the same methods, but the methods can behave differently based on the value of the instance variables.
* You can pass values into methods.
* A parameter is a local variable.
* Methods can return values.
* When method is of type void, it does not return anything back.
* We can declare the method to give a specific type of value back.
* Methods can have multiple parameters which have to be separated with commas when we declare them and separate the arguments with commas when we pass them.
* When a method has parameters, we must pass arguments of the right type and order.
* We can pass variables into a method as long as the variable type matches the parameter type.