# End of chapter 4 Notes

## 1) Overloaded methods (and constructors)

Method names and constructors that appear more than once in a class,
The parameter lists differentiate them.

## 2) Classes and objects

A class is a definition or a blueprint for a type of object.
An object is an instance of a class.
You can instantiate many objects from one class.

```
Coin c1 = new Coin();
Coin c2 = new Coin();
  ↖     ↖        ↖
class  objects   class
```

## 3) private versus public (access modifiers)

Members of a class - refers to the collection of instance variables and methods.

private members are invisible to clients of a class

public members are visible and accessible by clients of a class.

## 4) Encapsulation - a principle of object oriented programming

- all instance varibles should be private
- public methods should be provided for controlled access to objects' instance variables,

## 5) Variable Scope

**Local scope** - variables declared in a method or constructor.
Parameter declarations are also local variables.
They are only visible from within the method or constructor.

**Global scope** - instance variables have global scope
They are visible by the whole class; all methods and all constructors can see them.

* if a local variable has the same name as a global variable, the local variable gets precedence.

## 6) .toString()

Every class has a .toString() method, whether you supply it or not.

The default .toString() returns $\text{ClassName@}\overset{\text{object's address in memory}}{\text{3FA2B1}}$

The purpose of .toString() is to return a human readable representation of an object.

```
System.out.println(S1);   // will automatically do
                          // System.out.println(S1.toString());
```
$\uparrow$ an object