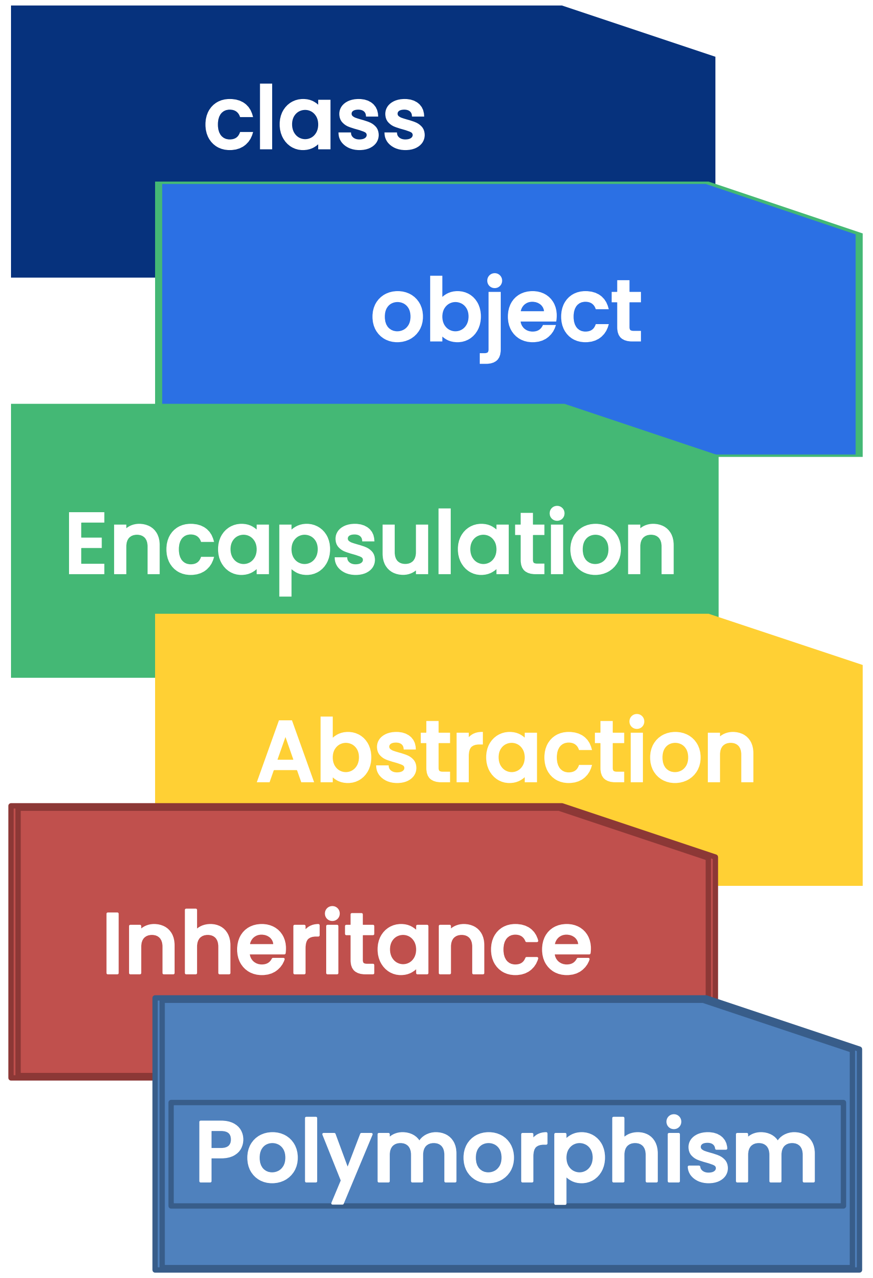
JA111

Day-3

Object Oriented Programming



The ultimate objective of object oriented programming is to present the things in computer as they are observed in the real world.

It is to bridge the gap between human thinking and computer

Class & object

A Class is the blueprint from which individual objects are created.

It is a passive entity.

It is a user defined data type.

It is a template that unites data and code together into a single entity i.e. this is called basic unit of encapsulation.

An object is an instance of a class that contains fields defined in the class and exhibit behavior.

It is a real world entity.

An object stores its state in fields (variables in some programming languages) and exposes its behavior through methods.

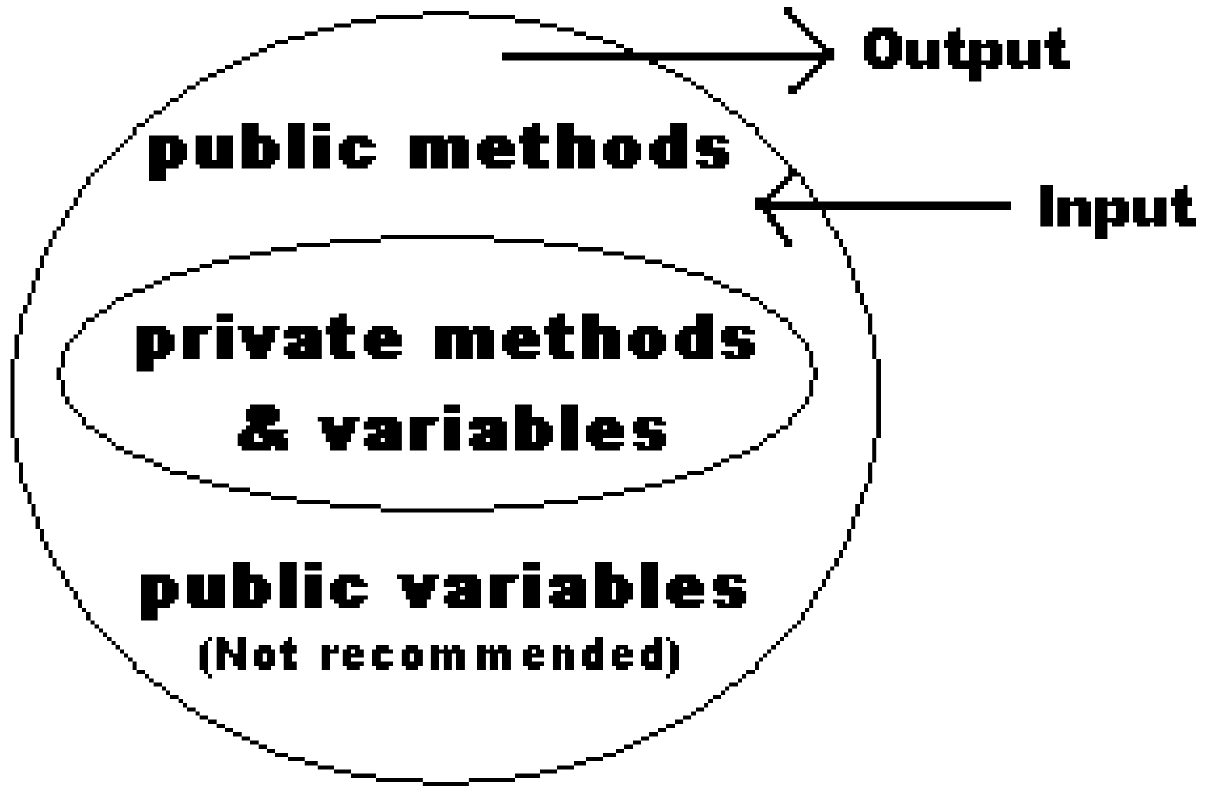
An object for class is what a variable for primitive type is.

Abstraction & Encapsulation

Abstraction refer to using an object without knowing its internal mechanism.

The wrapping up of data and its associated function together in a single unit (called class) and keeps both safe from outside interference and misuse is known as encapsulation.

Data hiding is an integral part of encapsulation which is achieved using access modifiers



Defining class and object in java

Defining classes

```
[access-modifier] class class-name{
   //field, constructor and method
declarations
}
```

Can you differentiate between static and dynamic memory allocation?

Can we assign one object to another? If yes then what impact will it have?

Defining objects

```
class-name obj-name; ---(1)
obj-name = new class-name(); ---(2)

class-name obj-name = new class-
name(); --- (1) + (2)
```

To access members of class-

```
obj-name.instance-variable-name;
```

Objects are created at run time but primitive data are allocated at compile time. Why?

Method

Method refers to a block of code that is used to perform a specific operation. Method increases modularity of the code.

```
[access-specifier] return-type method-name(parameter list) {
        //body of method
}
```

To call method of a class

```
obj-name.method-name(arg-list);
```

Primitive data types are always call by value hence changes made in parameter does not affect arguments.

Array, objects are passed by reference hence changes made in parameter affects arguments.

Polymorphism

- The word “poly” means many and “morphism” means forms i.e. polymorphism means many forms
- It is a feature which allows one interface to be used for general class of action and specific action is determined by nature of situation.
- It can be defined as "single interface, multiple methods"

Compile time

The behavior to be executed will be decided by the compiler i.e. which method will be executed is decided at the compile time that’s why it is Compile time polymorphism

- **Impl: Method Overloading**

Run time

The behavior to be executed will be decided by the run time i.e. which method will be executed is decided at the run time that’s why it is Run time polymorphism

- **Impl: Dynamic Method Dispatch**