Initializing A Jagged Array

==========

1. int [ ] [ ] arr=new int [3][ ];
   arr[0]=new int[3];
   arr[1]=new int[5];
   arr[2]=new int[4];

   arr[0][0]=10;
   arr[0][1]=20;
   .....

2.
   int [ ] [ ] arr=new int [3][ ];
   arr[0]=new int[]{10,20,30};
   arr[1]=new int[ ]{40,50,60,70,80};
   arr[2]=new int[ ]{90,100,110,120};

3.
   int [ ] [ ] arr=new int [ ][ ]{{10,20,30},{40,50,60,70,80},{100,110,120}};

4.
   int [ ] [ ] arr={{10,20,30},{40,50,60,70,80},{100,110,120}};

OBJECT ORIENTED PROGRAMMING

OOP is not a programming language. Rather it is a methodology (approach) to design programs which are close to the real world and help us visualize real world situations in our software

The biggest advantage of OOP is :

1. Data Security

2. Easy Modeling of real world situations

What are two basic building blocks of OOP?

The most impt components of OOP are:

1. Classes
2. Objects

Class: A class is just like a blue print or template which describes how an object will look.

For example: The model of a building developed by a Civil Engineer is like a class while the actual building builtup by the labourers is OBJECT.

In simple terms we can say a class is logical while objects are physical.

What does a class contain?

A class contains two impt elements

1. Instance variables : They are also called data members in languages like C++ and they hold the actual data stored in an object

2. Methods: These are operations which an object can perform. In simple terms they are functions associated with an object.

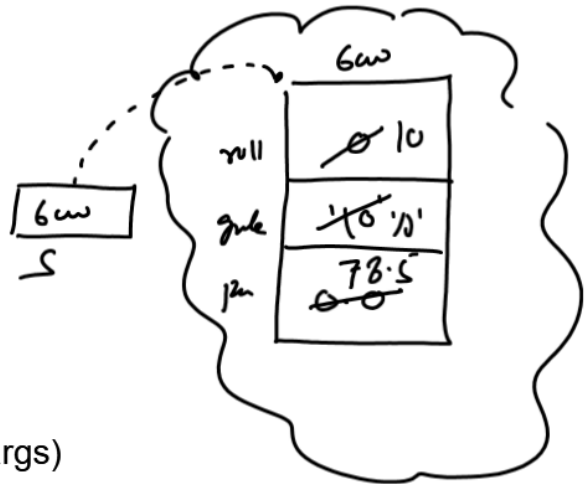
OBJECT
=======

As mentioned above a class is a logical unit so it never physically exist in memory. Thus in order to use a class we have to instantiate the class. The word instantiation simply means creating object of the class.

Syntax Of Defining a Class In Java
==========

```
<access modifier>  class <class_name>
{
   <access mod> <data type> <var>=value;
   <access mod> <data type> <var>=value;
   <access mod> <data type> <var>=value;

   <access_mod> <return_type> method_name( <arg>)
   {
        // body
   }
   <access_mod> <return_type> method_name( <arg>)
   {
        // body
   }

}
```

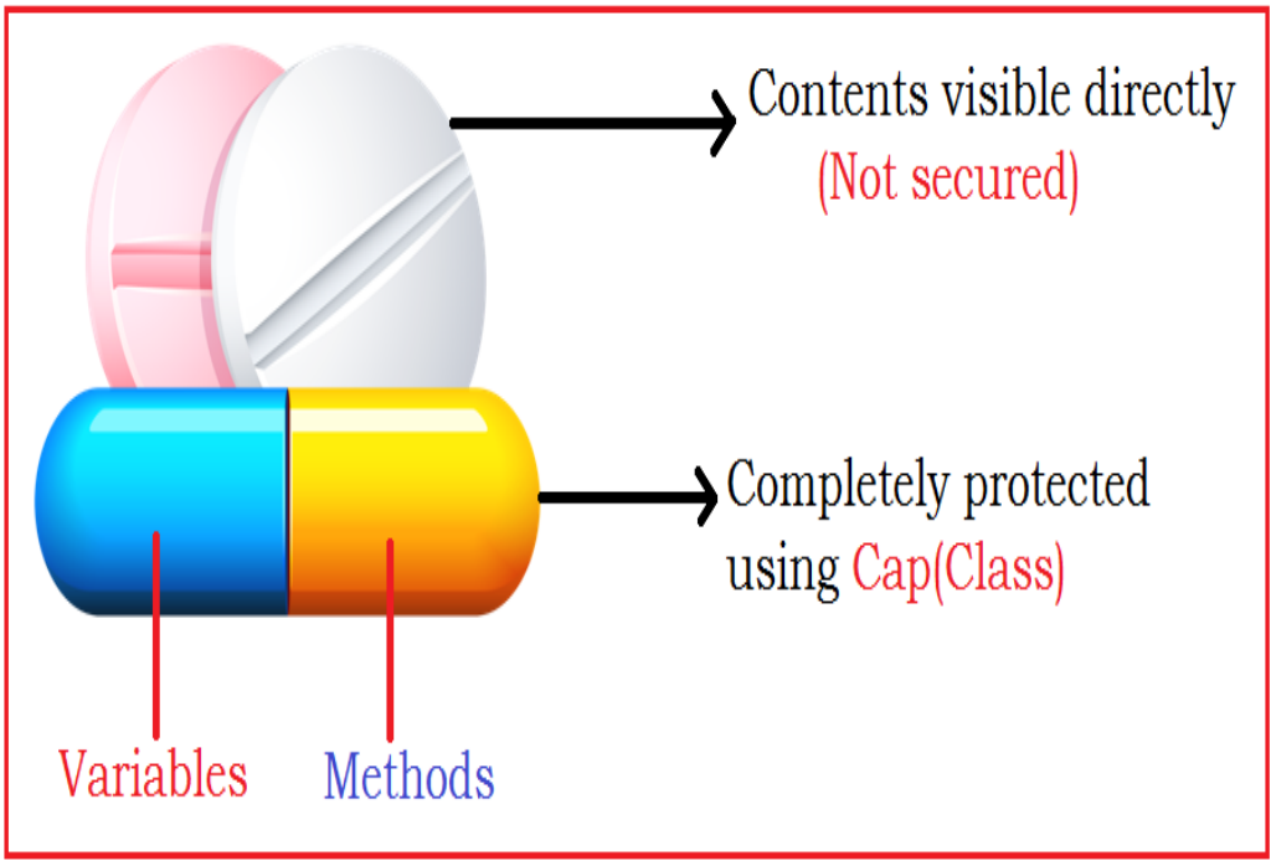
```
class Student
{
    int roll;
    char grade;
    double per;
}

class UseStudent
{
    public static void main(String [ ] args)
    {
            Student S;
            S=new Student();
            S.roll=10;
            S.grade='A';
            S.per=78.5;
            SOP(S.roll+","+S.grade+","+S.per);
     }
}
```



Draback with the previous code
==========
Although the above code is successfully compiled and executed but it it breaking a very impt principle of OOP called as **ENCAPSULATION**

What is Encapsulation?
==========
Encapuslation is one of the most impt principle of OOP which is derived from a word called **CAPSULE.** As we know a capsule represents mixture of multiple medicines packed in a single unit, similarly every program is a collection of two elements **data and functions .** The word data represent variables (intance variables in Java) and the word function represents methods. Since both data and functions always work together so OOPs says that they must be declare together also.

Thus declaring data and methods within a class as a single unit is called as **ENCAPSULATION.**



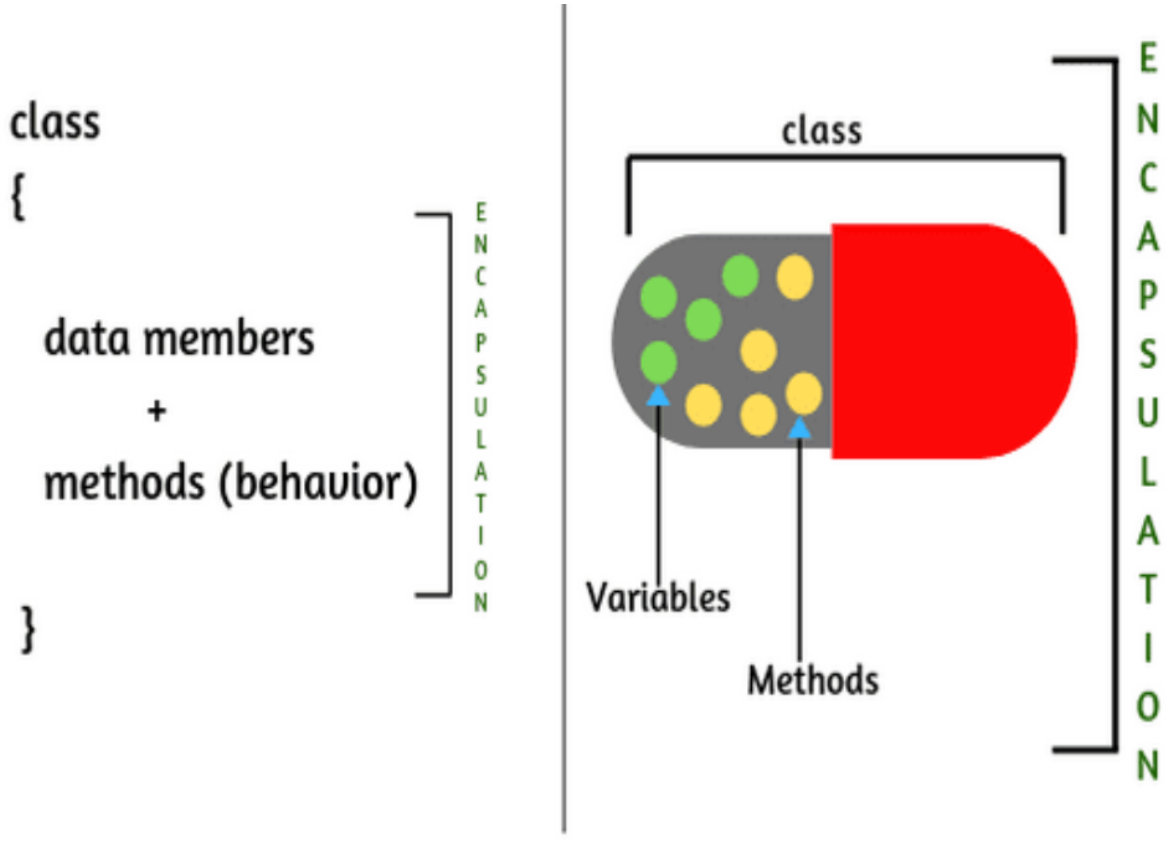
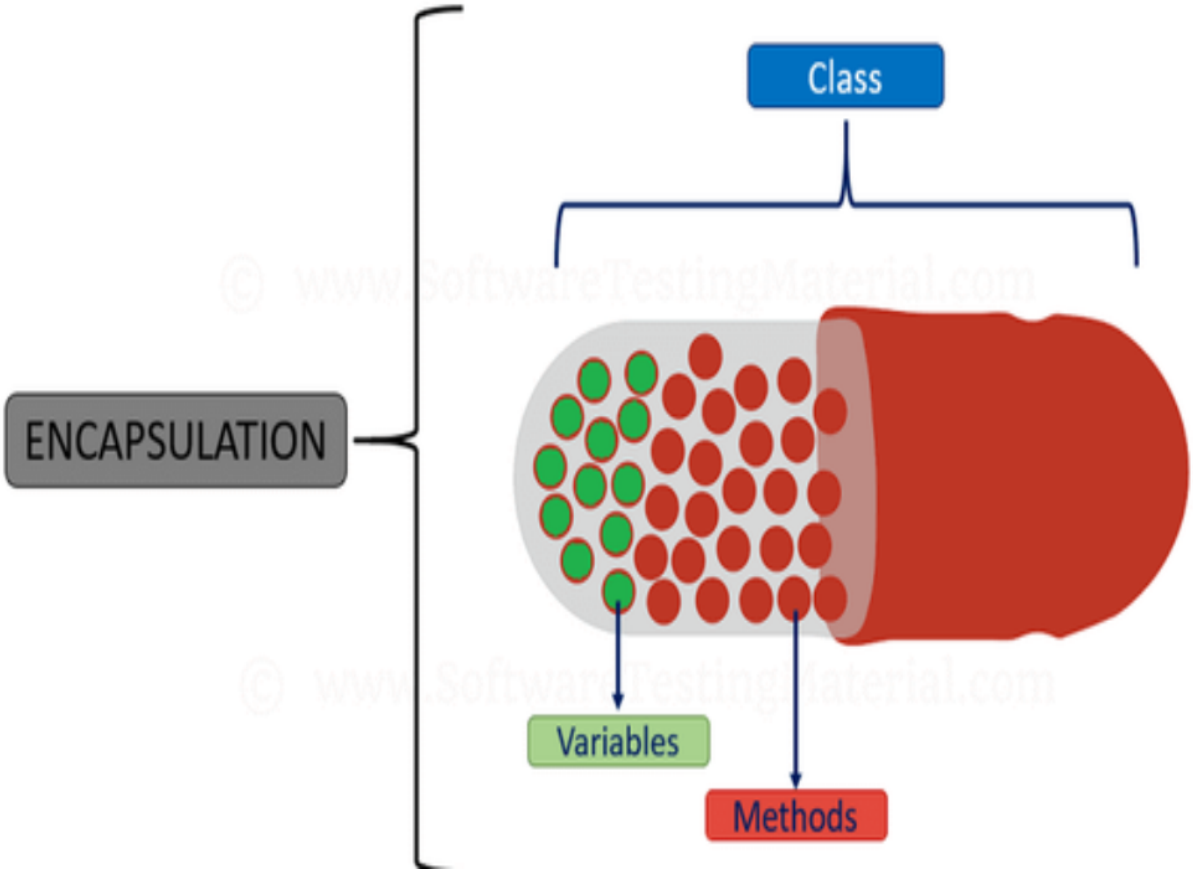


Fig: Encapsulation



Benefit

======

The most impt benefit offered by encapsulation is data security. This is because we can declare data members (instance variables in Java) as **private** and provide **public** method to access them. Due to this these data members can't be accessed directly from outside the class and will remain safe and secure.

How to acheive Encapsulation in Java?

==========

In Java to acheive encapsulation we have to take two steps while designing the entity class:

1. Every Instance variable must be declared as **private**
2. For every operation we must provide a **public** method in the class