**What is Data Hiding ?

Data hiding is nothing but declaring our non-static fields with private access modifier so our data will not be **accessible** from outer world (Outside of BLC class) that means no one can access our data directly from outside of the class.

*We should provide the accessibility of our data through **methods** so we can perform VALIDATION ON DATA which are coming from outer world.

//Program

```
package com.nit.data_hiding;

public class Customer
{
  private double balance = 10000; //Data Hiding

  public double getBalance()
  {
      return balance;
  }

  public void deposit(double amount)
  {
      //Data Validation
      if(amount <=0)
      {
          System.err.println("Deposit amount cannot be negative OR zero!!!");
          System.exit(0);
      }

      balance = balance + amount;
      IO.println(amount + " has deposited successfully...");
  }

  public void withdraw(double amount)
  {
      if(amount > balance)
      {
          System.err.println("In-sufficient Balance!!!");
          System.exit(0);
      }

      balance = balance - amount;
      IO.println(amount+" has withdrawn successfully");
  }

}
```

```
package com.nit.data_hiding;

public class DataHiding
{
   void main()
   {
       Customer raj = new Customer();
       IO.println("Current Balance is :"+raj.getBalance());

       raj.deposit(50000);
       IO.println("Current Balance is :"+raj.getBalance());

       raj.withdraw(40000);
       IO.println("Current Balance is :"+raj.getBalance());

   }
}
```

** Access modifiers in Java :

* An access modifier describes the **accessibility level of a class and the member of a class.**

* In order to define the accessibility level, Java has provided 4 kinds of access modifiers :

1) private access modifier [Within the same class]
2) default (package-private) access modifier. [Within the same package]
3) protected access modifier [Within the same package and from outside of the package using Inheritance]
4) public access modifier [No restriction, We can access from everywhere]

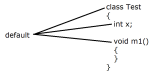
private access modifier :

It is the most restrictive access modifier because the member declared as private can't be accessible from outside of the class.

In Java we can't declare an **outer class** as a private or protected or static. Generally we should declare the non-static fields with private access modifier [Data hiding].

In java outer classes can be declared as public, abstract, final, sealed and non-sealed modifier only.

default (package-private) access modifier :



It is an access modifier which is **less restrictive** than private. It is such kind of access modifier whose physical existence is not available that means when we don't specify any kind of access modifier before the class name, field name or method name then by default it would be default.

As far as its accessibility is concerned, default members are accessible within the same folder(package) only. It is also known as package-private modifier.

protected access modifier :

It is an access modifier which is less restrictive than default because the member declared as protected can be accessible from the outside of the package (folder) too but by using inheritance concept.

Program : We are creating two java programs in two different packages :

com.nit.blc package

```
package com.nit.blc;

public class Test
{
    protected int x = 900;
}
```

com.nit.elc package

```
package com.nit.elc;

import com.nit.blc.Test;

public class Main extends Test
//Inheritance
{
   void main()
   {
       Main main = new Main();
       IO.println(main.x);
   }
}
```

Field x is accessible from another package because we are using Inheritance concept

public access modifier :

It is an access modifier which does not contain **any kind of restriction** that is the reason the member declared as public can be accessible from **everywhere** without any restriction.

According to Object Oriented rule we should declare the classes and methods as public whereas fields must be declared as private or protected according to the requirement.

Note: If a method is used for internal purpose only (like validation OR Logic calculation) then we can declare that method as private method. It is called Helper method.

⇩

```
public void makePayment(String name, String creditCard, double amount)
{
  if(validateCard(creditCard))
  {
  }
}

public void makePayment(String debitCard, double amount)
{
  if(validateCard(debitCard))
  {
  }
}

private boolean validateCard(String card)  //Helper Method
{
  if(card.length() !=16)
  {
    return false;
  }
  return true;

}
```

Diagram :

Access Modifiers	Within the same class	Within the same Package	From Another Package	From everywhere [No restriction]
private	YES	NO	NO	NO
default	YES	YES	NO	NO
protected	YES	YES	YES [using Inheritance]	NO
public	YES	YES	YES	YES

How to print Object Properties (non static fields) by using toString() method :