OOP CONCEPT IN JAVA

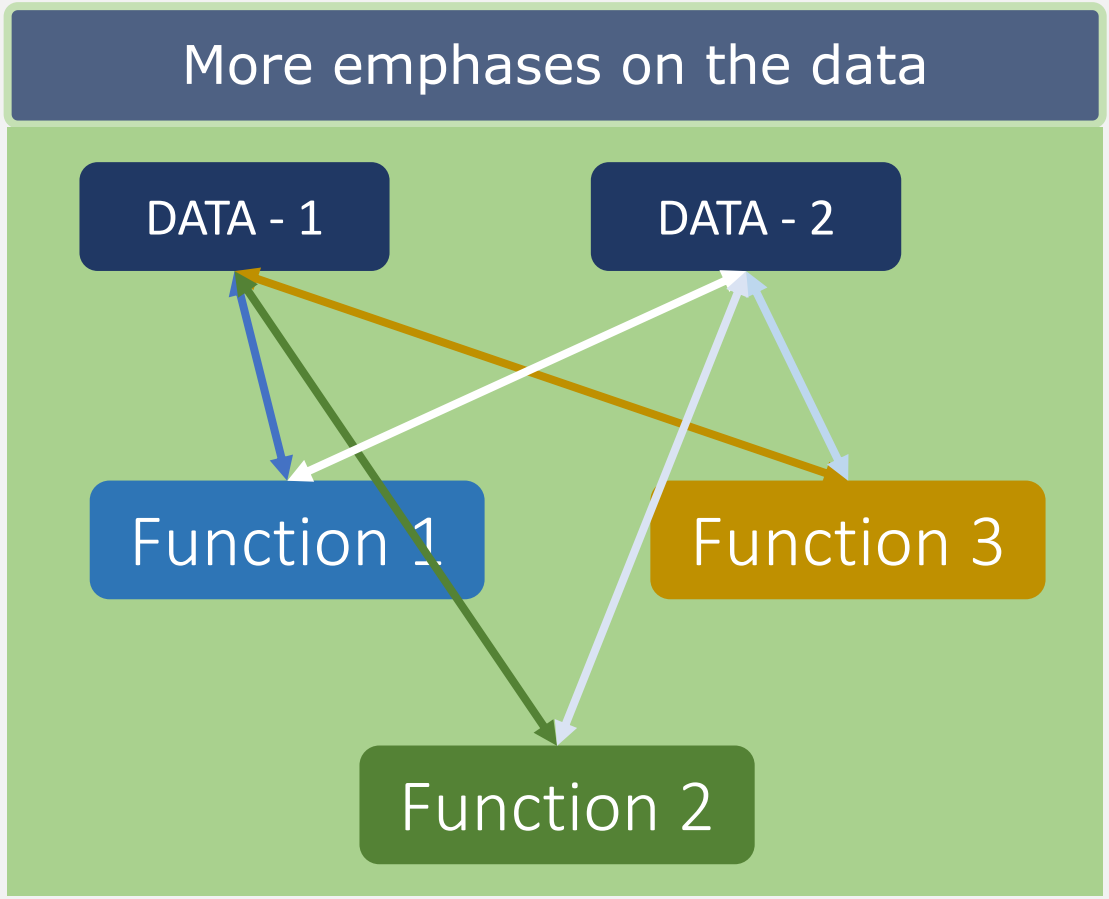
E-mail : **asif.io.edu@gmail.com**

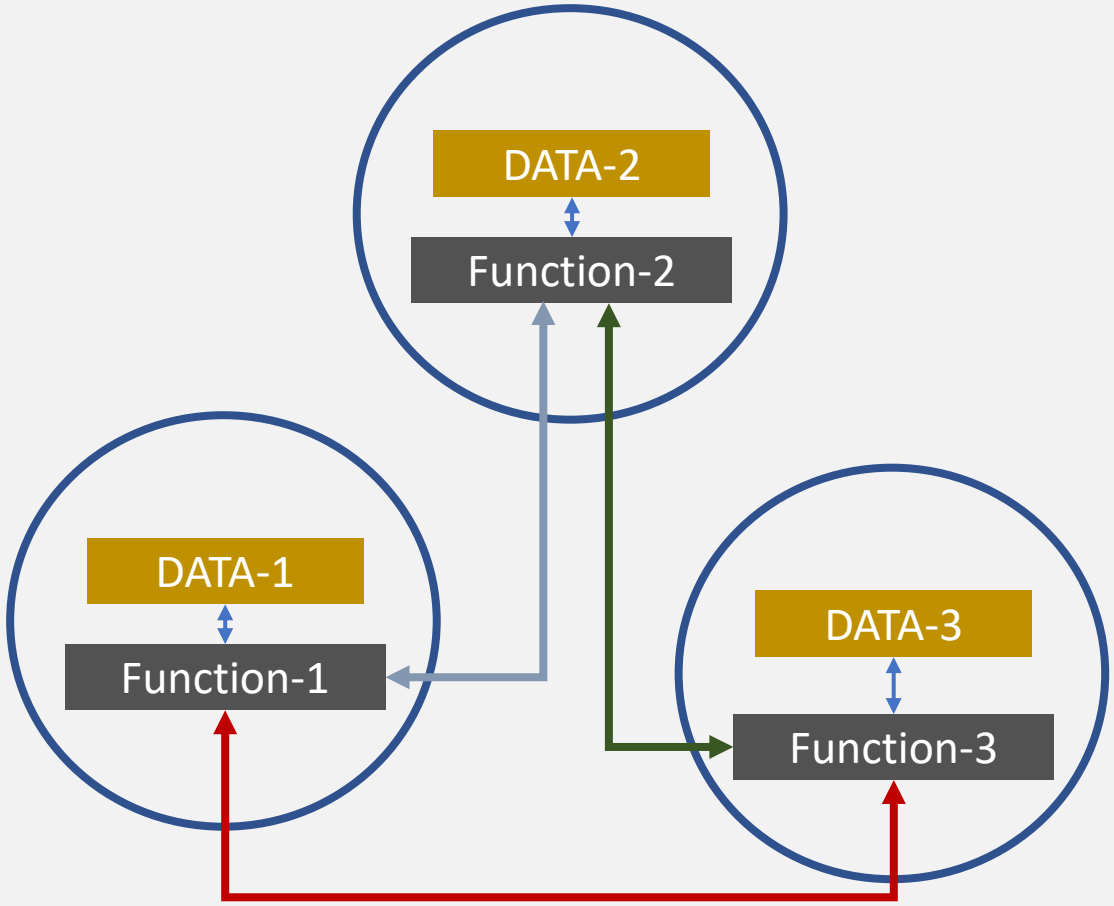
Requirement

Object Oriented Programing in concept in which more emphases are give in **function/method** rather than **data**.

Requirement

Object Oriented Programing in concept in which more emphases are give in function/method rather than data.





ABSTRACTION

POLYMORPHISM

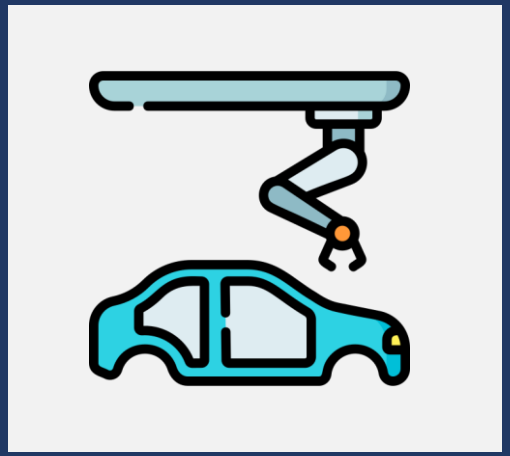
INHERITANCE

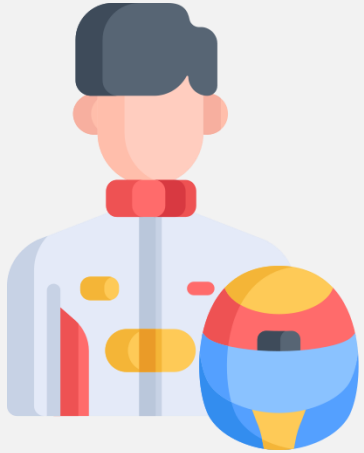
ENCAPSULATION

ABSTRACTION

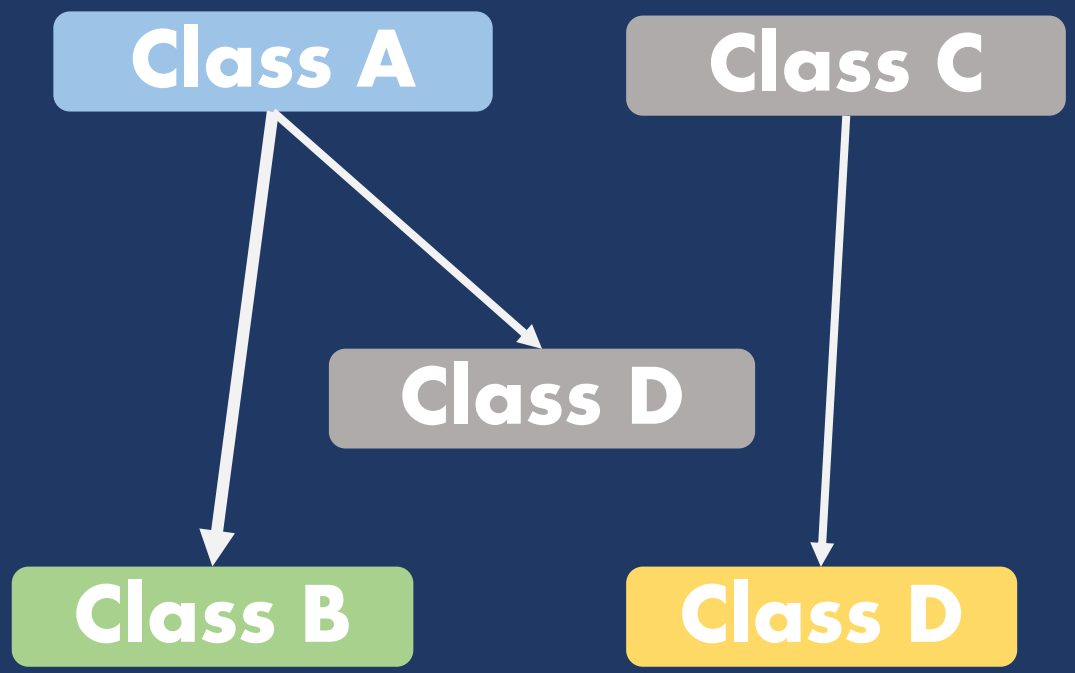
Data Abstraction is the property by virtue of which only the essential details are displayed to the user.

The trivial or the non-essential units are not displayed to the user.





INHERITANCE



Inheritance is the mechanism in java by which one class is allowed to inherit the features(attributes and methods) of another class.

- 👉 The parent class is known as base class.
- 👉 The children class is known as driver class.
- 👉 Multiple inheritance is not allowed in java. Means a children class can't have more then one parent class.
- 👉 "extend" keyword is used to implement inheritance in java.
- 👉 Child class override the method of parent's class in java.

E-mail : asif.io.edu@gmail.com

POLYMORPHISM

Polymorphism means one block of code can perform different task.

Poly = many
Morphism = form

One man but different role
(Father, Husband, Son, Brother)

SON

WIFE

MOTHER

SISTER

E-mail : asif.io.edu@gmail.com

Polymorphism - Code

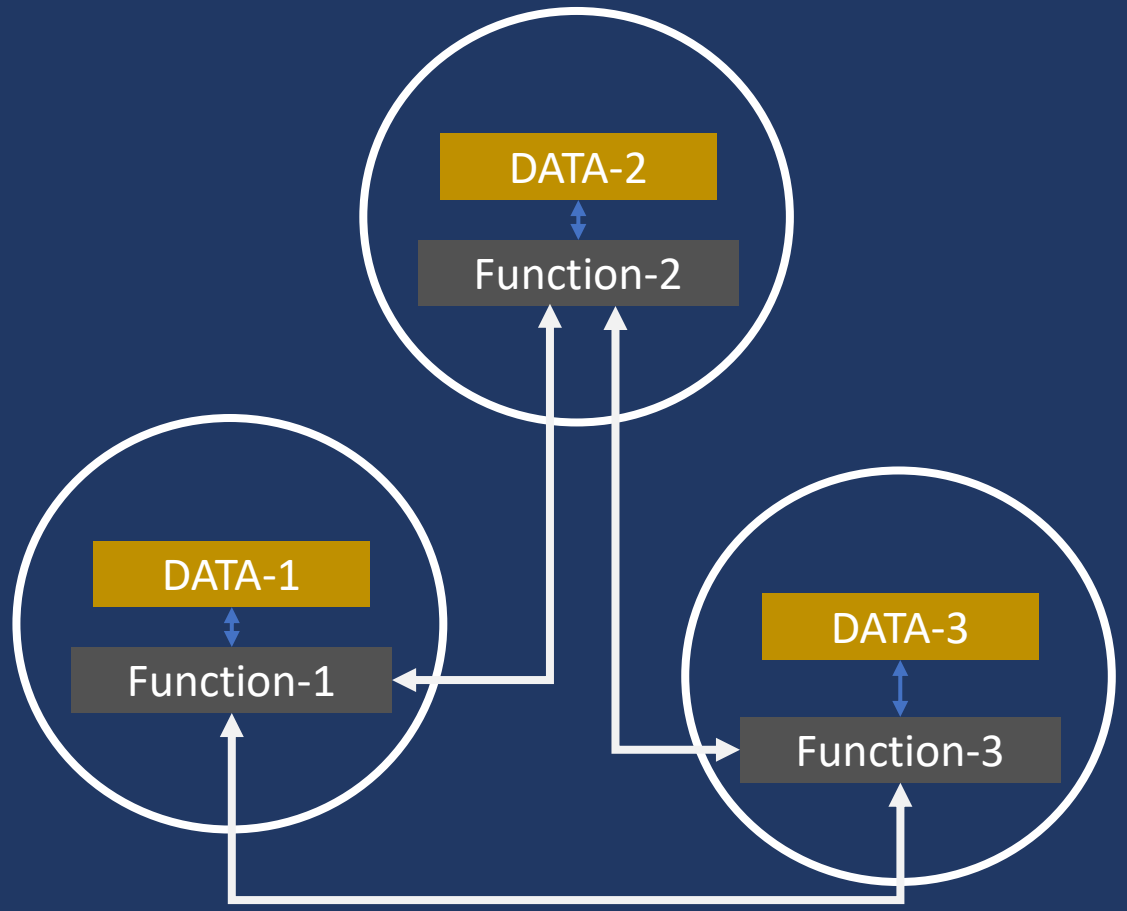
```java
public class Intro {
    public static void main(String[] args) {
        //Three different types of arrays.
        String[] name = {"Asif","Nasima", "Zeeshan","Anam"};
        Integer[] roll = {21,43,56,23};
        Character[] grade = {'A','B','D','A'};

        //Single methods printing three different Arrays
        printArray(name);
        printArray(roll);
        printArray(grade);
    }

    //Generic Method -- (example of polymorphism)
    //One method is used for printing three different type of arrays.
    3 usages
    public static <Thing> void printArray(Thing[] arr){
        for (int i = 0; i < arr.length; i++) {
            System.out.println(i+"-->"+arr[i]);
        }
        System.out.println("------------------");
    }
```

```
0-->Asif
1-->Nasima
2-->Zeeshan
3-->Anam
------------------
0-->21
1-->43
2-->56
3-->23
------------------
0-->A
1-->B
2-->D
3-->A
------------------
```

ENCAPSULATION



Encapsulation in Java is a process of wrapping **code** and **data** together into a **single unit**

👉 We wrap the attributes and function into a single unit.

👉 We make all the attributers private so the other class can't access them.

👉 Then we can create getters and setters so that we can access those attributes only through the functions.

👉 This technique of securing data is known as encapsulation.

E-mail : asif.io.edu@gmail.com

Encapsulation - Code

```java
public class Intro {
    public static void main(String[] args) {

        //Initializing object by using constructor
        BluePrint ct01 = new
                BluePrint( nam: "Himanshu", age: 17, ph: "42356776");
        BluePrint ct02 = new
                BluePrint( nam: "Saloni", age: 15, ph: "0987654345");

        //Accessing the data using Function
        ct01.getDetails();
        ct02.getDetails();

    }

}
```

```java
public class BluePrint {
    //these are attributes
    2 usages
    private String name;
    2 usages
    private int age;
    2 usages
    private String mob;

    //The 'this' keyword refers to the current
    // object in a method or constructor.
    2 usages
    BluePrint(String nam, int age, String ph){
        this.name = nam;
        this.age = age;
        this.mob = ph;
    }

    //Wrapping data and function in single unit
    2 usages
    public void getDetails(){
        System.out.println(this.name);
        System.out.println(this.age);
        System.out.println(this.mob);
        System.out.println("====================");
    }

}
```

E-mail : asif.io.edu@gmail.com

Java by ASIF.IO

E-mail : asif.io.edu@gmail.com