JAVA

UNIT 1
OBJECT ORIENTED PROGRAMMING CONCEPTS

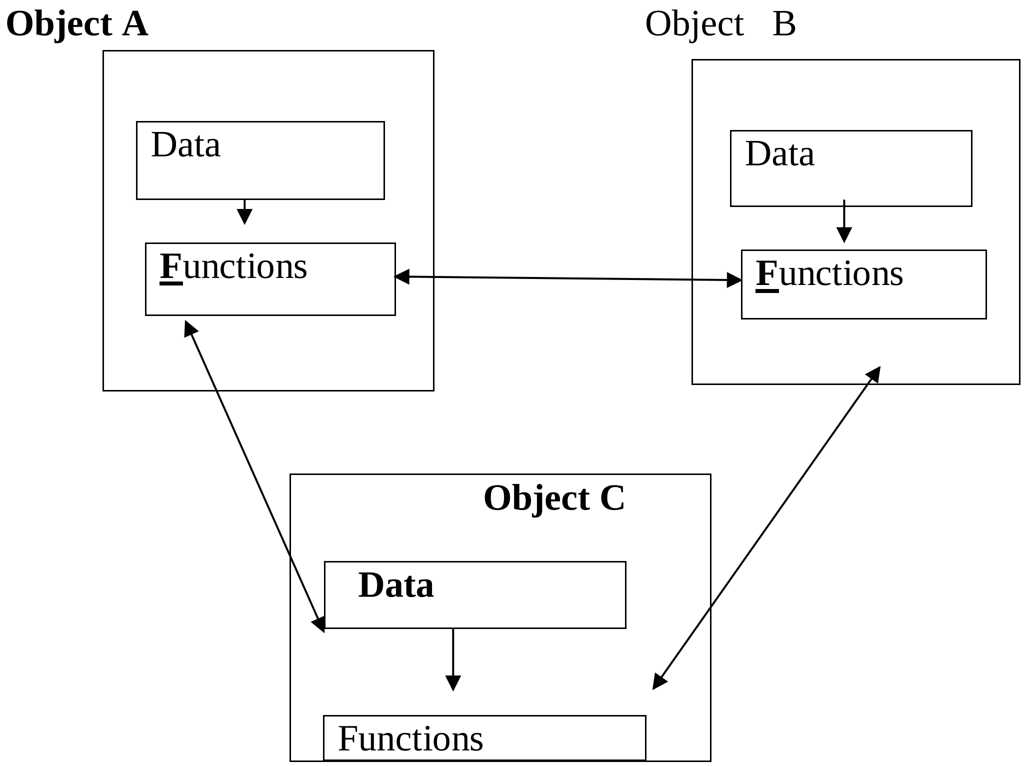
1.1 Know about object oriented programming

“object oriented programming as an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions “.

OOP treat data as a critical element .In object oriented design, problem is broken into objects .We write programs to serve the people in their day to day life. Therefore, programs should affect daily activities and services we use such as books, hospitals and railway reservation etc.

OOPs allows decomposition of a problem into a number of entities called objects and then builds data and functions around these objects.

The organization of data and functions in object oriented program is



Some of the characteristics of OOPS are:

- Emphasis on data rather than procedure.
- Programs are divided into objects.
- Data is hidden and cannot be accessed by external functions.

- Objects may communicate with one another through functions.
- New data and functions can be added whenever necessary
- Follows ‘bottom –up approach’ in program design.

Applications of OOPS:

- Real time systems
- Object – oriented databases
- Hypertext and hypermedia
- AI and expert systems
- Office automation systems etc.

1.2 Compare procedure oriented programming and object oriented programming:

Procedure oriented programming	Object oriented programming
1)Main task will classified into subtasks those tasks represented in the form of functions .	1) Main task will classified into subtasks those tasks represented in the form of objects .
2) Use top – down approach .	2) Use bottom–up approach .
3) No Acces modifiers in pop .	3) Acces modifiers in oop .
4) Data can be access freely from one method to another method .	4) In oop generally every thing take care by the access modifier .
5) Data hiding is not available in pop. So less secure .	5) Data hiding is available in oop. So more secure .
6) No method overloading .	6) We have method overloading .
7) Weakly typed language .	7) Strictly / Strongly typed language .
8) Debugging is difficult .	8) Debugging is easier .
9) Not suitable for larger programs and applications.	9) Suitable for larger programs and applications.
10) Faster	10) Slower
11) Data and procedure based .	11) Object oriented .
12) Only data and procedures are there.	12) Inheritance , encapsulation and polymorphism are the key features .
13) Less reusable .	13) More reusable .
14) Ex:- C ,Basic , Fortran .	14) Ex:- Java, C++ , VB.NET , C# .

1.3List and explain features of object oriented programming .

Properties of oops:_There are several fundamental concepts in object-oriented programming. They are

1) **Objects**
2) **class**
3) **Data abstraction and encapsulation**
4) **Inheritance**
5) **Polymorphism**
6) **Dynamic binding**
7) **Message passing**

1. Objects:

Objects are the basic runtime entities in an object oriented system. They may represent a person, or a place, or a bank account, Program objects must be choosen such that they match closely with real-world objects.

Each object contains code and data to manipulate the data. One way of representing an object

2. Classes: A class is a collection of objects of similar type for ex: mango, apple and orange are members of class fruit. The entire set of data and code of an object can be made a user-defined data type with class

Once a class has been defined we can create any number of objects belonging to that class. If fruit has been defined as a class, then the statement fruit mango; will create an object mango belonging to that class fruit.

3. Data Abstraction and Encapsulation

Data Encapsulation: "The binding up of (combining up of) data and methods into a single unit(i.e class)", is known as encapsulation.

The data is not accessible to the outside world, and only those functions which are wrapped in the class can access it. This concept is called data hiding.

Data Abstraction: "Abstraction refers to the act of representing essential features without including the background details or explanations."

4. Inheritance:

"Inheritance is the process by which objects of one class can acquire (get) the properties of objects of another class".

(OR)

It can be also defined as deriving a new class from already existing class.

- The main aim of inheritance is **"reusability".**
- The class(new class)which have properties of another class is called as "derived class".
- The class(old class)which provides properties is called "Base class".

Ex: Bird is a Base class
Attributes: legs, feathers, wings, nose
Flying bird or non flying bird are derived classes
Parrot is a derived class of flying bird and derived class of bird.

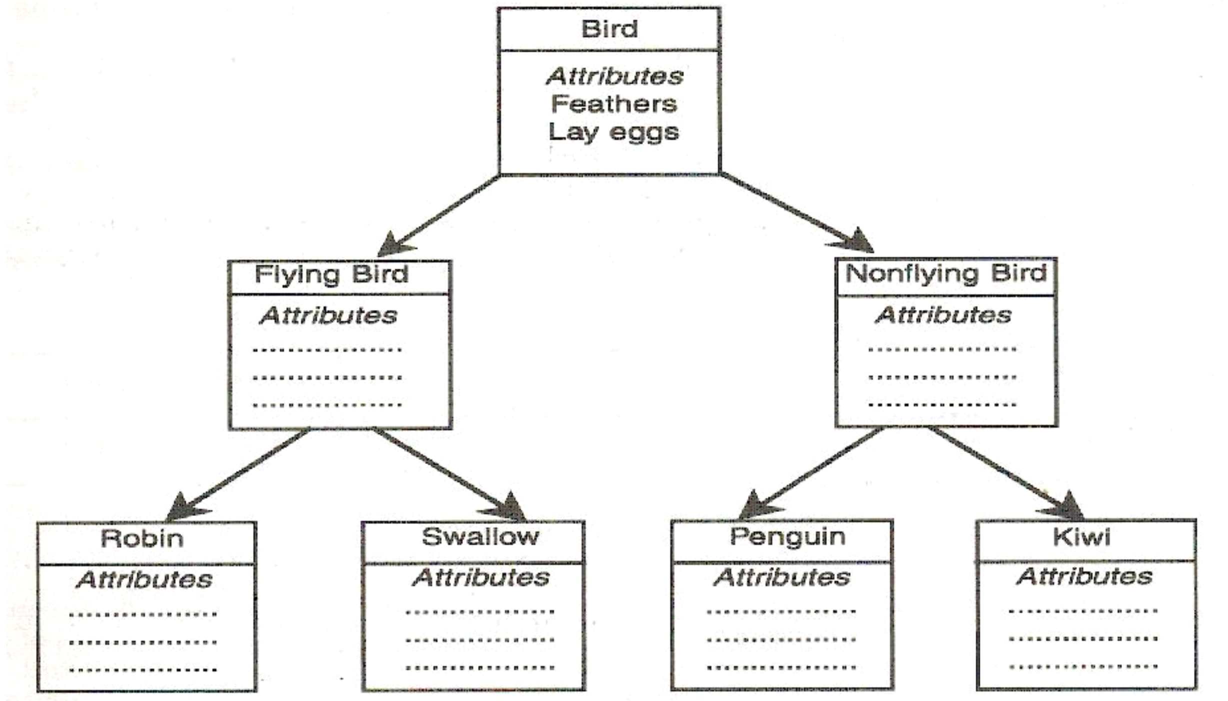


Fig. 3.1 *Property inheritance*

Types of inheritance:-

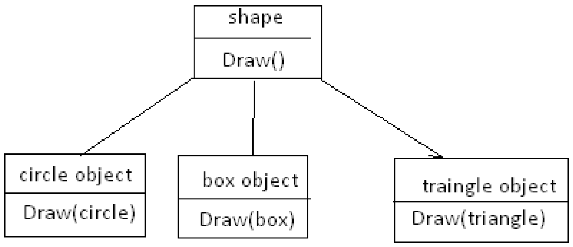
1) Single inheritance
2) Multilevel inheritance
3) Hierarchical inheritance
4) Multiple inheritance
5) Hybrid inheritance

5.Polymorphism:

"Polymorphism means the ability to take more than one form". An operation may exhibit different behaviors in different instances.The behavior depends on type of data. Polymorphism allows one name to specify a general class of actions

Ex: shape is one class
draw() is one function in draw object
draw(rect) draws rectangle
draw(circle) draws circle e.t.c.

fig: POLYMORPHISM



Polymorphism in java can be two types

- Compile time polymorphism
- Runtime polymorphism

Compile time polymorphism:-

Example: - method overloading

Method overloading : In java it is possible to create methods that have the same name but different parameter lists . This is called method overloading .

```
 import java.io.*;
 class Test
 {
 void m1()
 {
 System.out.println("this is m1 method");
 }
 void m1(int a , int b)
 {
 System.out.println("this is m1 method");
}
 void m1( char ch)
 {
 System.out.println("this is m1 method");
 }
 Public static void main(String args[])
 {
   Test t = new  Test();
    t.m1(10);
    t.m1(10,20);
    t.m1(‘a’);
```

```
        }
        }
```

Runtime polymorphism:-

Example :- method overriding

Method overriding:- If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in java**.

In other words, If subclass provides the specific implementation of the method that has been provided by one of its parent class, it is known as method overriding.

```
import java.io.*;

class A

{

void m1()

{

System.out.println("this is base class method");

}

}

class B extends A

{

void m1()

{

System.out.println("this is derived class method");

}

        Public static void main(String args[])
        {
            B b= new  B();
            b.m1();
        }
        }
```

6. Dynamic Binding:

Binding refers to the linking of a procedure call to the code to be executed in response to the call. Dynamic binding means that the code associated with a given procedure call is not known until the time of the call at runtime. Dynamic binding is also known as **late binding.**

Ex:- draw() procedure

At run time the code matching the object will be executed.

7. Message Passing:

An object oriented program consists of set of objects that communicate with each other.

The process of programming in an object – oriented language involves the following basic **steps:**

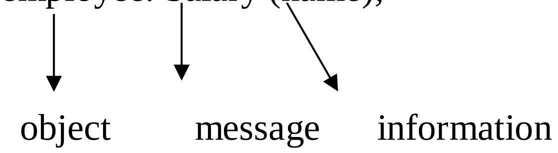
- Creating classes that define objects and their behavior
- Creating objects from class definitions
- Establishing communication among objects

Syntax:

Objectname.fucntionname(parameters);

Example:

employee. Salary (name);



1.4 Define class and object

Class:- Class is defined as the blue print for an object . A class serves as a plan (or) a template .Description of a number of similar objects is also called objects . Just defining a class does not create an object . we need create object explicitly. Classes are logical in nature .

For example furniture do not have any existence but chairs and tables do exist . A class is a user defined data type . A class contains data members(or)instance variables and methods .

syntax:-

```
Class  classname
 {
```

```
                [ fields declarations] //instance variables(or)datamembers
                [ method declarations]
             }
```

Every thing square brackets is optional .

Ex :-

```
    Class Rectangle
  {
       int length,breadth;
           void getdata(int x,int y)
              {
                length  =x;
                breadth =y;
              }
           int  rectarea()
            {
                 int    area=   length  * breadth;
                  return(area);
            }
}
```

Object:-

Object is an instance of a class**.**

Ex :- table , chair are all instances of a class furniture . objects can be physical . objects have unique identity , state and behaviour .

An object in java is essentially a block of memory that contains space to store all the instances variables . Creating an object is also refered to as instantiating an object .

Objects in java are created by using a keyword new operator . the new operator creates an object of the specified class and returns a reference to that object.

Syntax:- classname objectname;
Objectname = new classname();

Example :-

```
          Rectangle rect1;
         rect1 = new Rectangle();
```

Both statements can be combined into one as shown below :

```
          Rectangle rect1 = new Rectangle();
```

<u>1.5Compare class and C structure</u>

CLASS	C STRUCTURE
1.Class is a collection of data members and methods.	1.Structure is a collection of data members or structure members
2.In class members are public/private/protected.	2.In structures members are public.
3.In class members are by default private.	3.In structure members are by default public.
4.Class is a reference type datatype.	4.Structure is a value type datatype.
5.Object is created in the heap memory.	5.Variables is created in the stack memory.
6.Class can be inheritied.	6.We can not inherit the structure.
7.Object should be created using "new" keyword. Syntax: ClassnameObjectname=newclassname();	7.It is not mandatory.
8.Class is created by using a keyword **'class'**.	8.Structure is created by using a keyword **'struct'**.
9.It occupies more space.	9.It occupies less space.

10.Syntax:
Class classname
{
Instance variables;
Methods();
}
	10.Syatax:
	Struct tagname
	{
	Data member1;
	-----------------;
	Datamember n;
	};

11. Example:
class Rectangle
{
int length, breadth;
viod getdata(int x, int y)
{
Length=x;
Breadth=y;
}
}
12.The member variable of a class can be initialized directly.
	11. Example :
	struct student
	{
	int id;
	char name[20];
	int sub[3];
	};
	12.The member variable of structure cannot be initialized directly.