SYNERGY SCHOOL OF ENGINEERING
DHENKANAL

OBJECT ORIENTED METHODOLOGY

SEM:3RD

LECTURER NOTE

PREPARED BY
SMRUTI MAYEE MISHRA
DEPT:CSE

Programming Language

Q: What is programming language?

A programming language is a computer system of notation for writing computer programs.

→ A programming language is a computer language that is used by programms to communicate with computers.

→ Its mainly used to develop desktop application, websites and mobile application.

Types of programming language:-

Generally there are three types of programming language that is :- (i) Low level language
(ii) Middle level language
(iii) High level language

(i) LOW level language :- (Machine level language)

→ low level / machine level language is also known as binary language (0/1). which computer directly understand and no translater is used there.

→ Machine language is a set of instruction executed directly by the central processing unit.

[program is a set of instruction]. (CPU)

→ Each instruction performs a specific task such as load, jump or ALU (Arithmatic logic unit) operation (logical operation).

(ii) MIDDLE LEVEL LANGUAGE:

→ It bridges the gape between machine level language

and high level language.

→ It helps in writing system programming as well as application programming.

EX:- C, assembly language:

High level language:-

→ Its easy to convenent and write the program by putting less effect.

→ But High level language cant be directly understand by the computer, so translater is used here.

→ Translater is ~~used here~~. like, compiler or interpreter is used to perform to convert high level language to low level language.

EX:- C++, Java, python, (Net).

→ High level language are divided into two parts.

(i) procedure oriented programming (POP) language

(ii) object oriented programming (OOP) language

(i) Procedure oriented programming language:-

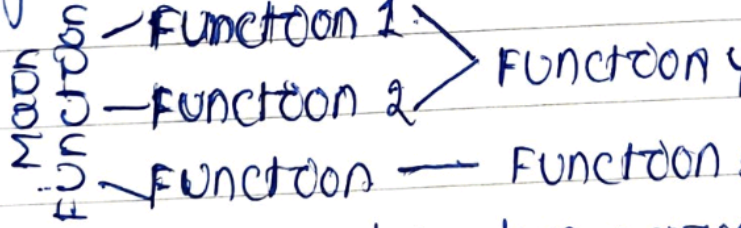
→ In POP language are devided into smaller programme / procedures. This program is written as a sequence of procedure (function).

→ Each procedure contains a series of instructions for performing a specifire task. During the program execution each procedure can be called by other procedure.

→ To call a procedure we have to write procedure name only.
→ The major emphasis of these language is on the procedure but not only the data.
→ POP language allow the data to move freely around the system.
→ For sharing of data among multiple function many important data items are declare as Global. This makes data moves openly around the system from function to function., * function may transform data from one form to another.

DIAGRAM :-

→ program structure of POP :-



→ POP language is used top down approach.
EX :- FORTAN ; COBOL, BASIC, C and ALGO.

Relationship of data and function in POP :-

→ In POP global data can be access and changed by any procedure/functions. There is no data security. In case if we want to change type of data global data then we also need to update all functions that access the data. Due to this it ~~mainly~~ may happend some error.

→ To call a procedure we have to write procedure name only.

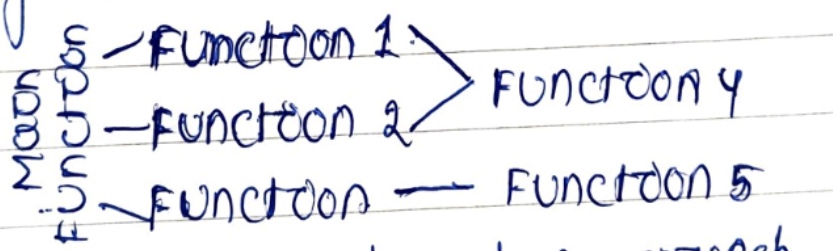
→ The major emphasis of these language is on the procedure but not only the data.

→ POP language allow the data to move freely around the system.

→ For sharing of data among multiple function many important data items are declare as Global. This makes data moves openly around the system from function to function. * function may transform data from one form to another.

DIAGRAM :-

→ program structure of POP :-



→ POP language is used top down approach.

EX :- FORTAN, COBOL, BASIC, C and ALGO.

Relationship of data and function in POP :-

→ In POP global data can be access and changed by any procedure/functions. There is no data security. In case of we want to change type of data global data then we also need to update all function that access the data. Due to this it ~~mainly~~ may happend some error.

protection its from accidental notification from outside functions.

- Its protects the data on the outside.
- The oop data is hidden (secure) and can't be access by extends functions.
- In oops object communicate with each other through functions.
- In oops emphasis of data rather than procedure
- oops follow bottom - up approach

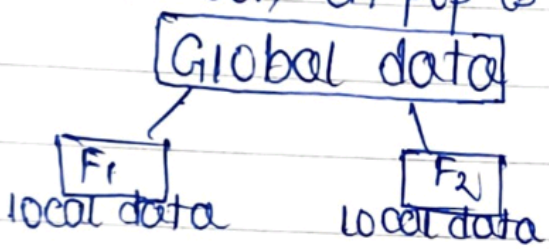
Main function

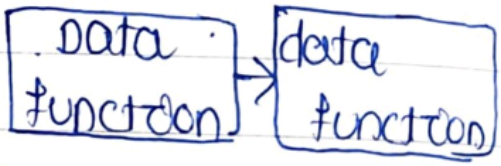
F1 F2 F3 ↑ bottom of approach

Difference between pop & oop :-

pop	oop
(i) In pop is devided into smaller program is called procedure.	(i) In oop program is denoted into smaller entities called object.
(ii) It follows top to down program design approach.	(ii) It follows down - top program design approach.
(iii) Importance given to the algorithms rather than data	(iii) Importance given to data rather than algorithm.
(iv) In pop data move openly around the system.	(iv) In oop data can't move openly around the system. Data hidden is used in oop.

(v) It doesn't model real world problem very well.	(v) OOP model real world problem.
(vi) It creates medium size projects.	(vi) It creates large size projects.
(vii) In POP the complexcity of the program is very high.	(vii) In OOP the program complexcity is very less.
(viii) POP doesn't have any proper way for holding the data. So it's less secure.	(viii) OOP provide proper way for hidden the data so it is more secure.
(ix) Relationship of data function in POP is.	(ix) Relationship of data and function in OOP is object A





(x) Ex :- C, COBOL, FORTRAN, BASIC PASCAL.	(x) Ex :- C++, JAVA, VB.Net, PYTHON.

Basic concept / characterstics / Features of OOPS :-

→ There are generally 6 features of OOPS i.e.

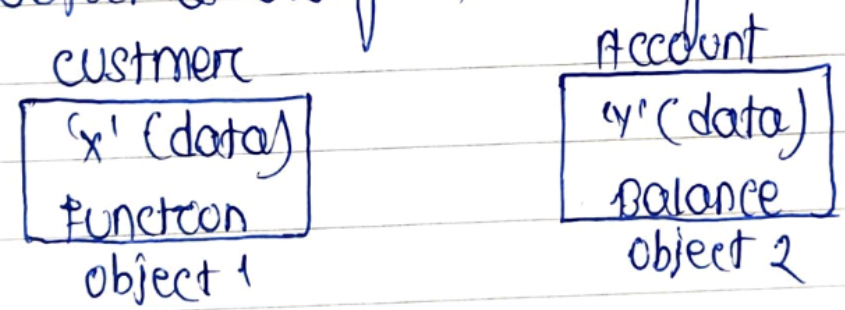
1. objects
2. class
3. Data Hiding
4. Data Encapsulation
5. Data Abstraction
6. Inheritance

⑦ polymorphism
⑧ Dynamic Binding
⑨ Message passing

① objects :-

→ object are basic run time entities in oops system.
→ object may be a person, a place, a bank A/c, a table of data, or any items that program handle.
→ objects can represent the real world problem.
→ object occupy space in memory.
→ During program one object can communicate with another object through message passing
object is always Activeentity

EX :-



y balance (x)
object 1 — Information
Message passing

② class :- (collection of similar objects)

→ class is a collection of similar objects.
→ class is used to create user define data types.
→ It behaves like built in data type in programming language

CLASS → Objects → DATA → UDDT (user defined data type)
Always passvantity

→ It behaves like data type
→ It alongate the memory for storage.
→ class is a blue print of an object that contains variables for storing data and function to perform operations on these data.
→ once a class has been defined we can create any number of object belong to that class.
→ It is a logical representation, so there is no memory is reserved.

EX :- suppose fruit is a class.

FRUIT (class)
Name -
shape -
colour -
seed qualities :-

Mango (object)
Name - Mango
shape - circular
colour - yellow
seed qualities :-

Representation of class :-

student

class :-
data - name age Roll no.
% age of attendance Test ()

syntax of a class :-

class is devided into 3 parts : 1- private
2- public
3- protected

```
class student -> class name
    {
public : data member;
         member function ();
private : data member;
          member function ();
protected : data member;
          member function ();
```

Data Hiding :-

- Data cannot be accessed directly by the outside is called data hiding.
- Data hiding will be done by using private access specification.

Data encapsulation :-

- The wrapping of data and function into a single unit is known as encapsulation.
- Data encapsulation can be achieved by using class.
- By data encapsulation, data is not accessable to the outside class only these function which are wrapped in the class can access it.
- Function of the class provide the interface between

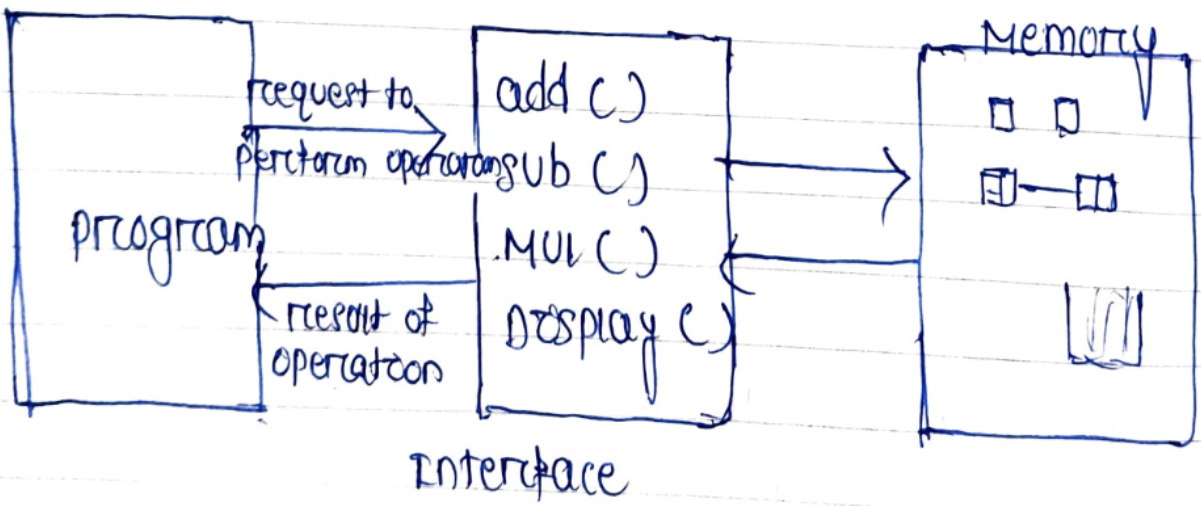
the objects data and outside objects or function.

→ Difference between encapsulation & data hiding :-

Encapsulation	Data hiding
(i) Encapsulation concerns about wrapping data to hide the complexity of system.	(i) Data hiding concerns about data along with hiding complexity of system.
(ii) Encapsulation focus on wrapping the complex data.	(ii) It focus on restricting or preventing the use of data inside the capsule.
(iii) It may be public or private.	(iii) It always private.

Data Abstraction :-

→ It refere to the act of representing essential part and hide the data of the programming. as known as data abstruction.



Inheritance :-

- Inheritance is a mechanism of containing the features and behaviour of a class by another class.
- The class whose members are inherited is called the base class and the class that inherits those member is called the derived class.
- Inheritance implements the 'IS-A' Relationship.
- In this process object of one class contains the properties of object of another class.
- Inheritance is the concept of oop in which one class inherits the data (attribute) and method of another class.
- The class whose properties and methods are inheritade is known as parent class and the class that inherits the properties from parent class is called child class.

Types of inheritance :-

1. single inheritance
2. Multilevel inheritance
3. Hierarchical inheritance
4. Multiple inheritance
5. Hybrid inheritance

1. single inheritance :-

When a class inherits another class is known as single inheritance.

class A

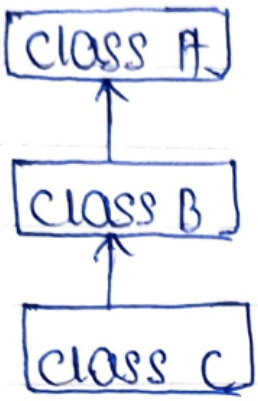
↕

class B

```
class Animal
  {
void eat ( )
  {
system.out.println ("eating");
  }
class dog extends Animal
  {
void bark ( )
  {
system.out.println ("barking");
  }
class Test 1
  {
public static void main (system arg [ ] )
  dog d = new dog ( )
      d.bark ( );
      d.eat ( );
        }
          }
```

② Multilevel inheritance :-

When there is a chain of inheritance it is known as multilevel inheritance



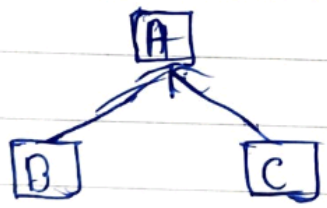

```
class Animal
  {
void eat ( )
  {
system.out.println("eating");
  }}
class dog extends Animal
  {
void bark ( )
  {
system.out.println("barking")
  }}
class baby dog extends dog
  {
 void cuddle ( )
  {
```

```
system.out.println("cuddling");
        }}
class Test 2
    {
public static void main (system arg [])
    {
baby dog b = new baby dog ( )
        b.bark ();
        b.eat ();
        b.cuddle ();
        }}
```

EX:- baby dog class inherits the dog class which again inherits the animals class so there is a property of multilevel inheritance

③ Hierarchical inheritance :-

When two or more classes inherite a single class, it is known as Hierarchical inheritance



EX: Dog and cat are two classes which inherits from the animal class so there is a Hierarchical inheritance

```
class Animal
    {
void eat ()
    {
system.out.println("eating");
    }}
```

```
class dog extends Animal
  {
 void bark ()
  {
system.out.println ("barking");
    }}
 class cat extends Animal
  {
  void meow ()
  {
system.out.println ("meowing");
    }}
 public static void main (system arg[])
  {
 cat c= new cat ()
   c.eat ();
   c.meow ();
 dog d.= new dog ()
   d.eat ();
   d.bark ();
    }}
```

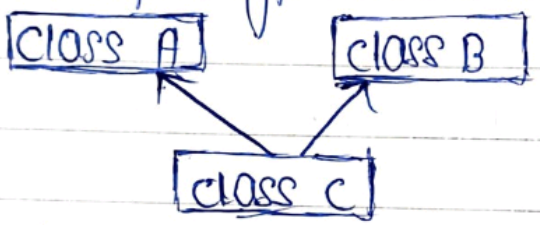
(4) Multiple Inheritance :-

→ Multiple inheritance is not supported by JAVA.

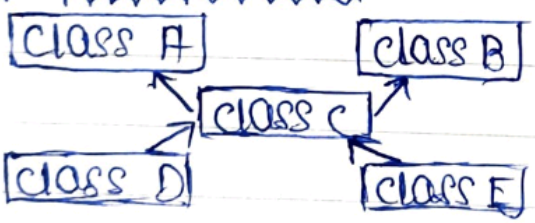
→ To reduce the complexity and simplify the language multiple inheritance is not supported in JAVA.

→ consider a scenario where a, b and c are three classes. The class inherits from a and b classes have the same methods and you call it from child class object so that there will be ambiguity to call the method a and b class. so that compile type error will be find out

Diagram :-



(5) Hybrid Inheritance :-



Defination :-

Hybrid combination of more than one inheritance is known as hybrid inheritance

Multiple inheritance program EX :-

```
class A
{
void msg()
{
```

```
system.out.println("Hellow");
    }
    }
class B
  {
void msg()
  {
system.out.println("Hi");
    }
    }
class C extends A,B
  {
public static void main(system argg[])
  {
  c.obj = new c();
   obj.msg();
    }
    }
```

polymorphism :-

→ polymorphism is a greek word which indicates "poly" means many and "morphism" means forms. so polymorphism means one name having many form.

→ It is defind as the ability of a message to be displayed in more than one form.

EX:- A person at the same time can have different characteristics. For example a man at the same time is a father, brother, husband and employ

→ It means same person posses different behaviour in different situation this is called as polymorphism.

→ There are two examples :-

① over-loading : abs ( ) is one method which can write abs(int), abs (float) and abs (long)

② over-riding :

→ same methods name but different argument types. which is known as overloading.

① over-loading :-

For example absolute is one method which can write

```
abs (int);
abs (float);
abs (long);
```

here it is same method name but different argument is known as overloading.

② over-riding :-

EX :-

```
class A
  {
  display ( )
  {
system.out.println ("Hello");
  }}
class extends A
  {
  display ( )
  {
```

```
system.out.println("Hi");
    }
  A a = new A()
    a.display();
  }}
```

→ Here the same method but multiple implementation is known as overriding.

DYNAMIC BINDING :-

→ Binding means collecting, a linking, of the method (function) body to the method call.

EX:-

```
void show()
  {
  system.out.println("Hellow");
  }
Text t = new Text();
  t.show();
}}
```

binding formation

(calling)

Binding are two types :- (i) STATIC BINDING
(ii) DYNAMIC BINDING

(i) STATIC BINDING :-

→ When type of the object is determined at the compile time is known as static binding.

EX:-

```
class dog
  {
```

(4) Multiple Inheritance :-

→ Multiple inheritance is not supported by JAVA.

→ To reduce the complexity and simplify the language multiple inheritance is not supported in JAVA.

Bind Formation

```
void eat ()
  {
  system.out.println ("eating");
  }
}
public static void main (system arg[])
    dog d = new dog ();
    d.eat ();
  }
}
```

(5) DYNAMIC BINDING :-

→ When type of object is determined at run time it is known as dynamic binding.

EX :-

```
class Animal
  {
  void eat ()
    {
  system.out.println ("eating");
    }
  }
class dog extends Animal
  {
  void bark ()
```

```
system.out.println("barking");
  }}
class test1
  {
public static void main(system arg[])
  {
  dog d = new dog();
      d.eat();
      d.bark();
    }}
```

output :- eating
barking

→ Here the object type can not be determined by the compiler because the instance of dog is also an instance of animal. so compiler does not know its type only it knows the base type.

MESSAGE PASSING :-

→ Message passing means communication between processes.

→ Here one object communicate with other object is called message passing.

→ There are two types of message passing method :-

① put message (function)

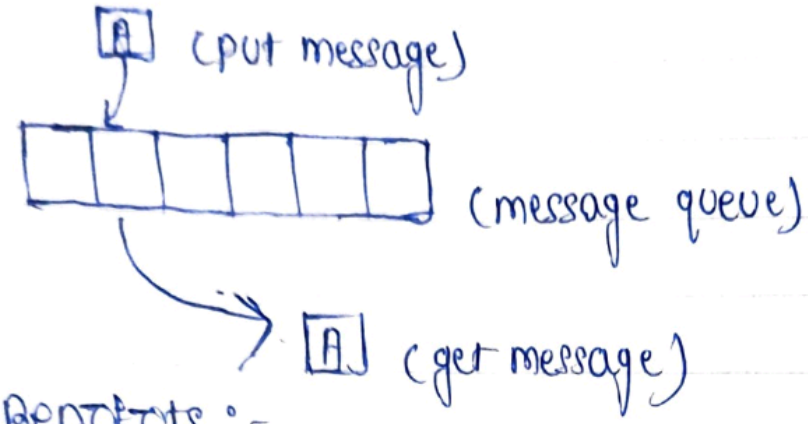
② get message (function)

① Put message () :-

Add a message in the message queue.

② Get message () :-

Extract the message from the message queue.



OOP Benefits :-

- Through inheritance we can eleminate / reduce redundance (duplicate) code. Extend the use of existing classes.
- The principle of data binding helps the programmer to build secure program that can't be invaded (hacking) by code in other part of the program.
- Its possible to have multiple objects to co-exist without any objection / interferance.
- Its easy to partition the work in a project based on objects.
- object oriented system can be easily upgraded from small to large system.
- Message passing technique for communication between objects make the interface description with external system is much simpler.
- software complexity can be easily manage.

Application of OOP:-

- Real time system → ~~some~~
  simulation and modeling
  object oriented data base

Hyper text; Hyper media and expert text
A.I (Artificial intelligence) and expert system.
Neural network and parallel programming (Multitasking).
Decision support and office automation system
cad. system

OVERVIEW OF JAVA :-

→ JAVA is a general purpose oop language
→ Generally java program is of two types.
(i) STAND ALONE APPLICATION
(ii) WEB APPLET

(i) STAND ALONE APPLICATION :-

→ Its a program which is written in JAVA to carry out certain tasks on stand alone local computer
→ For executing a stand alone java program involves two steps :-
(A) computing source code into byte code using JAVA 'c' compiler
(B) After that executing the byte code program using JAVA interpreter

(ii) WEB APPLET :-

→ It is a java program which is developed inside internet application.
→ Applet located on serverc computer through internet and executed on local computer (client computer) using a java browser.
→ We develop applets for doing everything from simple

animated graphics to complex games and utilities.

So applet add embedded in an HTML (Hyper text markup language) and run inside a web page, creating and running applet are more complex than creating an application.

HISTORY OF JAVA :-

JAVA is a general purpose object oriented programming language developed by sun microsystem of USA in 1991.

First it was named as 'OKK' by James Gosling. After 1995 again renamed as JAVA.

JAVA was designed for the development of software for electronic devices like TV, VCR, AC, Refrigerator, Toaster, washing machine etc.

The main role of java is to develop a reality, simple, reliable, portable, and powerfull language.

JAVA FEATURES :-

① Compiled and Interpreted :-

JAVA compiles combines both compile and Interpreted.

First it compiled and gives us byte code after that it will be interpreted to get the output. So that java is two stage system. (Important)

First java compiler translate source code to byte code, Byte code are not machine instruction, therefore in the second stage java interpreter generates machine

code, which is directly executed by the machine running inside java programming. Thus java is both compiled and interpreted language.

(2) platform independent and portable :-

JAVA program can be easily moved from one computer to another anywhere and anytime.

If there is a change in processor, operating system or system resource then there is no change in java program.

The byte code instruction that can be implimated on any machine.

The size of primitive data types are machine independent.

(3) object oriented :-

All program code and data are present within an objects and classes.

The object model in java is simple and easy to extend.

· Roboust and secure :-

JAVA is strictly check the data type.

JAVA has a concept of exception handly, which captures serices error and eliminates any risk of crashing system.

Internets has so many threats. so java system verify all memory access and also check that no virus can communicate with an applet. since java program can not access to memory locations without

proper authorisation.

Distributed :-

JAVA is a distributed language for creating applications on network. It enables and allows multiple programmers at multiple remote locations to colabrate and work together on a single project.

Simple, small and familiar :-

JAVA is small and simple language.

It eleminates some critical features of c and c++ like, it does not use pointer. pre-processer header files go to statements operator overloading and multiple inheritance etc

It is familiar because it is modelled on c and c++ language. Java uses many constructs of 'c and c++'.

Multithreaded and Interactive :-

Multithreaded means handling multiple tasks simultaneously

JAVA supports multithreaded program, It means it execute multiple programms or applications at the same time

DYNAMIC and Extensibility :-

JAVA is a dynamic language which is capable of dynamically linking in new class libraries methods and objects.

It also supports functions writing in other language c and c++, This functions are known as native method

This fascility enables the programmer to use

the efficient functions available in this languages.
Native methods are loved dynamically at run time

Scalability and performance :-
JAVA increasing in scalibility and performance improving startup time and reducing the amount of memory used in java at run time environment

Monitering and managibility :-
JAVA supports numbers of API (application program interface) such as JVM.
Monitering and management API. some management platform - extention loging, monitering and management interface and java management extension. (JME).

Difference between JAVA and C :-
The main difference between java and C is java is object oriented programming so that is uses object and class concept : JAVA does not include some 'C' features like :(i) JAVA does not contain the ~~data type~~ unique statement key words like size up and type.
(ii) It does not contain the data type struct and union
(iii) It does not define the time modifier key words like auto, extern, register, signed and unsigned
(iv) It does not support pointer.

(V) It does not have preprocessor, so that we can use # define, # include etc.

Difference between JAVA and C++ :-

JAVA does not support operator overloading.

JAVA does not support multiple inheritance of classes.

JAVA does not support global variable. Every variable and method is declared within a class and form the part of the class.

JAVA does not use pointer.

There is no borderline in JAVA.

JAVA environment :-

JAVA environment includes a large number of development tools and hundreds of classes and methods. Which is the part of java standard library (JSL). It is also known as (application programming interface).

The development tools are part of the system is known as JAVA development kit (JDK).

JAVA DEVELOPMENT KIT (JDK) :-

JDK is the collection of tools. It is used for developing and running JAVA program. It includes applet viewer it enables the java plate.

JAVA C - (JAVA COMPILER) :-

The java compiler translates java source code to byte code files. Which can understand by interpreter.

JAVA :-

JAVA interpreter runs applets and applications by reading an interpreting byte code files.

JAVA ah :-

It produces header files for use with native methods.

JAVA AP :-

JAVA disassembler which enables to convert byte code files into a program description.

JDB :- (JAVA DEBUGER) :-

JAVA debuger which helps us to find errors in our programms.

Text editor
↓
Java source code → Java document → HTML file
↓
Java c
↓
Java class file → Java → Header files
↓
Java → JDB
↓
Java program output

Process is :-

The user written programms is :-

Source code

compiler ↳ Byte code

Interpreter ↳ Machine level language

In the above diagram it is the process of building and running java application program.

To create a java program has we need to creat a source code file using a text editor. then the source code is compiled using the java compiler. (java c) and executed using the java interpreter.

If any error is find out then JDB (Java debuger) can detect it.

A compiled java program can be converted into source code with the help of java disassembler java ap<>

API (Application programming Interface)

Inside API there are several classes and methods. group into packages :- Java language support package

A collection of classes and methods required for implementing basic feature of Java

(ii) Utilities package :-

A collection of classes ~~and methods~~ to provide utility functions such as date and time function

(iii) Input / output package :-

A collection of classes required for input/output manipulation.

Networking package :-

A collection of classes for communicating with other computer through internet.

AWT package :- (Abstract window toolkit) :-

It contains classes that implements platform independent graphical user interface.

Applet package :-

This includes a set of classes that allows us to get java applet.

Java run time environment :-

(i) Java virtual machine (JVM) :-

It is a program that interprites the intermediate java code, byte code and generates the required outputs.

(ii) Run time class libraries :-

These are sets of core class libraries that require execution of java programs.

(iii) User interface toolkit :-

There are various tool kits is present by guests user can communicate easily

Simple java program :-

```
class Test
{
```

```
public static void main (string args[])
{
system.out.println ("Hello")
}}
```

How to save the file, compile and run the program :-

1 - open note pad and add the code
2 - save the file as Test.java
3 - open a commadd promote window and go to the directory where you saved the program.
4 - Type java c. Test.java and press enter to compile. If there is no error the command go for next line
5 - Now type Java .Test to run your program
6 - Now you will able to see the output "Hello".

Main function :-

Every java application program must include the main function method. This is the starting point for the interpreter to begin the execution of the program.

public :-

The key word public refers that it is an access specifier, that declars the main method so it is accesable to all other class.

static :-

It is a key word we were declars that method

belongs to entire class and not a part of any object of the class. The main function must be declared as static because the interpreter uses this method before any objects are created.

void :-

void states that main method does not return any value

system :- It is the inbuilt class.

out :- It is a member (object of a system class)

println () :- It is the method which gives us the output to the system.

print (-) :- It means after heading the output the cursor is present at the same output line but in println it go towards the next line.

Basic syntax :-

① case sensitive :-

Java is case sensitive so Hello and HELLO both are different meaning

② class name :-

For all class name the first letter should be in upper case

③ Method name :-

All method names should start with a small letter

④ Program file name :-

Name of the program file should exactly match the class name. When we are saving the class, we should save it by using class name and append

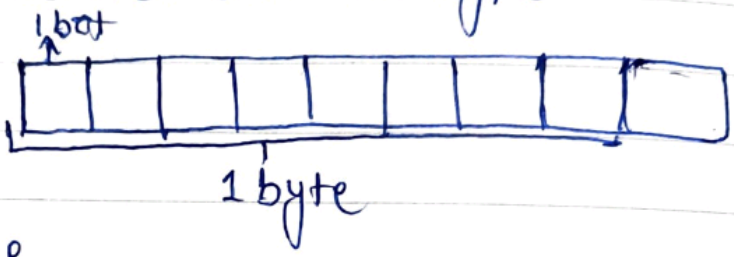
Java to the end of the file name.

Data types in JAVA :-

Data types specifying the different sizes and values that can be store in variable.

Data type is devided into two types.

1 byte = 8 bit



- Data type
  - Primitive Data type
    - Numeric data type
      - Integer
        - Byte - (1)
        - short - (2)
        - int - (4)
        - long - (8)
      - Decimal
        - float
        - double
    - Non-numeric
      - character (1 Byte)
      - Boolean (1 Bit)
  - Non-primitive datatype
    - Class
    - Array
    - String
    - Union
    - Structure

Writing style in Java :-

Byte b, c
short a,
int a,
Boolean a,

long - a
float - a
char - a
double - a

```
class Ax
{
public static void main (string args[])
{
boolean a,b
a= True
system.out.println ("a is" + " " +a);
b= false
system.out.println ("b is" + " " +b);
}}
```

Variable in JAVA :-

Variable is the name of the memory location which is given by the user. The variables can stores any type of value..

For storing and accesing the data we must know the name of the variables.

Variable is of 3 types :- (i) Local variable
(ii) static variable
(iii) Instance variable

(i) Local variable :-

A variable which is declare inside method of method parameter is called local variable

Ex :- void sum (int a)
{

int x ;

(ii) Instance variable :-

A variable which is declared a set of class but outside of the method is called instance variable

Ex :-
```
class A
{
int a;      // Instance of variable.
{
public static void main (string args [])
}}
```

(iii) static variable :-

A variable which is declare with the help of static key word is called static variable.

```
class A
{
static int a;   // static variable
{
public static void main (string args []):
}}
```

All variables example :-

```
class A
{
int a=10;
static b=20;
public static void main (string args[])
{
int c=30;
A ab=new A();
system.out.println(ab.a) -> for instance variable
system.out.println(A.b) -> for static variable
system.out.println(c) -> for local variable
```

Addition of two numbers:-

①

```
class A
{
public static void main (string args[])
int a=10;
int b=20;
int c=a+b;
{
system.out.println(c);
}
```

②

```
class A
{
```

```
int a = 10;
int b = 20;
int c = a+b;
public static void main (strong args[])
    A ab = new A ();
    {
system.out.println (ab.c);
    }
```

(Q) calculate the area of the rectangle.

Ans)
```
import java.util.*;
class Area
{
public static void main(String args[])
{
Scanner A = new Scanner(System.in);
~~int~~ system.out.println("Enter the value of length")
int length = A.nextInt();
system.out.println("Enter the value of breadth");
int breadth = A.nextInt();
int area;
area = length * breadth;
system.out.println("area = "+area)
}
}
```

Using class:-
```
import java.util.*;
class Area
{
int l, b;
void a ()
{
system.out.println(l*b);
}
}
```

```
class Test1
  {
public static void main (String args [])
  {
Scanner sc = new Scanner (System.in).
System.out.println ("Enter the value of l & b");
   sc.l = sc.nextInt();
   sc.b = sc.nextInt();
 Area c = new Area ();
     c.a ();
     }
      }
```

Method overloading :-

```
class Rectangle
  {
int l, b, area, perimeter;
  void A ()
  {
 area = l*b;
   }
 void p ()
  {
 perimeter = 2*(l+b).
   }
   }
```

```
class Rectangleap
{
public static void main (String args[])
{
Rectangle rec = new Rectangle ();
rec.l = 5;
rec.b = 6;
rec.A ();
system.out.println ("area =" + area);
rec.p ();
system.out.println ("perimeter =" + perimeter);
}
}
```

O/P = 30 22

Date :- 03-11-2023- Friday

parameter method overloading :-

Adding a parameterized method to a class :-

```
class Rectangle
{
int width, length, are;
void area ().
{
are = width * length;
}
```

```
void setvalue (int w, int l);
        {
        w = width;
        l = length;
        } }
public static void main (String args[])
        {
  Rectangle rec1 = new Rectangle ();
  Rectangle rec2 = new Rectangle ();
  rec1 . setvalue (10, 20);
  rec1 . area ();
 System. out. println(area1. ar);
     rec2 . setvalue (30, 40);
     rec2 . area ();
 system . out. println (rec2 . ar);
        }
```

CONSOLE :-

Its a predefined class ~~that is~~ that is available I/O package. It is use to get user input and runtime.

con ① Readline ( ); - string (return type is string)

② readpassword ( ); character (return type is character)

system console ( ) - Here console is a function or method.

import package - It is used for security purpose

syntex:- console objectname = system.console();

programme:-

```
import java.io.*;
class con
{
public static void main (String args[])
{
string a, character b[];
console obj = system.console();
system.out.println("username");
a = obj.readline();
system.out.println("user name:" + a);
b = obj.readpassword();
system.out.println("password:" + c);
}
```

For show password option ~~add = string c = string.~~
then, (system.out.println ("show password:" ~~value of (b);~~ + c)
or ("password is" + c))
(otherwise you declare the string c at first)

constructor:-

constructor is a special type of method whose name is same as class name.

Note :-

The main purpose of constructor is to initialize the object.

Every java class has a constructor.

A constructor is automatically called at the time of object creation.

A constructor never contain any return type including void.

Syntax :-

```
class classname
  {
   classname( )
    {
     }
  }
```

Program :-

```
class con
  {
 int a; string b
   con ( )
   {
  a=0, b="null"
   }
 void display ( )
   {
```

```
System.out.println (a+" "+b);
        }
class con1
    {
public static void main (String args[])
    {
  con d=new con ();
    d.display ();
      }
      }
class con
    {
int a, String b ;
    {
  void display ();
    {
  System.out.println (a+" "+b);
    }
    }
class con1
    {
public static void main (String args[]).
    {
  con d=new con ();
    d.display ();
      }
      }
```

O/P (output)
a=0
b=null

O/P (output)
a=0
b=null

Types of constructor :-

1. Default constructor
2. parameterized constructor
3. copy constructor
4. private constructor

1. Default constructor :-

A constructor which doesnot have any parameter then it is called as default constructor

syntax :-

```
class A
{
A()
{ }
}
```

Program :-

```
class A
{
int a, strong b, boolean c;
A()
{
a=100;
b="abcd";
c="true";
}
void show()
{
```

```
system.out.println (a+" "+b" "+c);
    }}
class B
  {
public static void main (string args[]).
  {
A a= new A ();
 a.show ();
  }}
```

Output

a=100
b=abcd
c=true

→ When we skip the part a=100, b="abcd", c="true" in this program (value) then the output is a=0
b=null
c=false

② parameterised constructor :-

A constructor through which we can pass one or more parameters is called parameterised constructor.

```
class A
  {
int a, int b, string c, int d;
  A(int x, int y)
    {
    a=x;
    b=y;
    }
```

```
A ( int x, string z)
  {
   d = x;
   c = z;
  }
 void display ()
  {
 System.out.println (a+ " " +b);
 system.out.println ( c+ " " +d);
  }}
 class B
  {
 public static void main (string args[])
  {
   A a1 = new A (10,20)
   A a2 = new A (30,40)
     a1.display ();
     a2.display ();
  }}
```

Copy constructor :- Date :- 06-11-2023, Monday

class

Whenever we pass object reference to the constructor then it is called as copy constructor.

syntax:- class classname
{
classname (obj.ref)
{
}}

program:-
```
class A
{
int a; string b;
A ()
{
a = 10;
b = "abcd";
system. out. println ("a:" + a + " " + "b:" + " " + b)
}
A (A ref)
{
a = ref.a;
b = ref.b;
system. out. println (a + " " + b);
}} class B
{ public static void main (string args [])
{
```

```
A s1 = new A ();
A s2 = new A (ref/s1);
    }}
```

private constructor :-

In java it is possible to mode a constructor as a private but according to the rule we cannot access private members outside of the class

syntax :- class classname

```
{
    private classname ()
    {
    }
}

class A
{
int a; string b;
private A ()
{
a = 10;
b = "abcd";
system.out.println ();
}
public static void main (String args [])
{
A s1 = new A ();
}
```

constructer overloading :-

```
class A
{
int a; string b;
A()
{
a=5;
b="abcd";
System.out.println("a="+a+" " +"b="+b)
}
A(int x)
{
a=x;
System.out.println(a);
}
A(int x, string y)
{
a=x;
b=y;
System.out.println(a+" "+b);
}}
class B
{
public static void main(string args[])
{
```

```
A ab=new A();
  ab.A();
  ab.A(10);
  ab.A(15, "gh");
    }}
```

Inheritance :-

Date:-11-10-2023

When we construct a new class from existing class in such a way that the new class access all the features and properties of existing class is called inheritance.

Note :-

In case of java the extends key word is used to perform inheritance it provides code reusability.

We can not access private members of class through inheritance.

A sub class contains all the features of superclass.

Method overloading only possible through inheritance so we should create the object of subclass.

Syntax :-

```
class Superclass
  {
   }

class subclass extends superclass
  {
   }
```

Types of Inheritance :-

There are 5 types of inheritance :-

(1) **Single inheritance :-** Date :- 08-12-2023 - Saturday

Single inheritance is the type of inheritance which contains only one superclass and only subclass is called as single inheritance.

Syntax :-
```
class superclass
    {
    - -
    }
class subclass extends superclass
    {
    - -
    }
```

Ex :-
```
class Add
  {
  public static void main(String args[])
    {
    int a=10, b=40;
    int add;
    add = a+b;
  system.out.println(add);
    }
  }
```

```
class A
  {
int a, b, c;
  void add():
 a=10; b=20;
   c=a+b;
system.out.println(c);
      }}
 class B extends A
    void sub()
     {
 a=20; b=10;
   c=a-b;
 system.out.println(c);
      }}
 class Result
     {
public static void main(string args[])
     {
 B b = new B();
   b.add();
   b.sub();
      }}
```

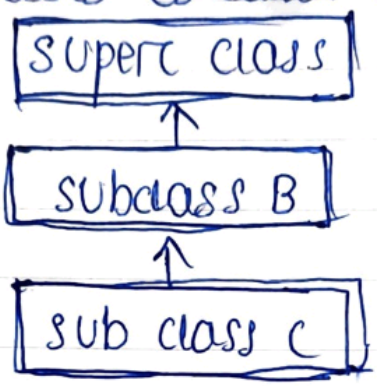
```
class A
  {
int a, b, c;
  void add():
a=10; b=20;
  c=a+b;
system.out.println(c);
    }}
class B extends A
    void sub()
      {
a=20; b=10;
  c=a-b;
system.out.println(c);
    }}
class Result
    {
public static void main(string args[])
      {
B b = new B();
  b.add();
  b.sub();
    }}
```

(2) Multilevel Inheritance :-

In multilevel inheritance we have only one superclass and multiple subclasses is called multilevel inheritance.

~~syntax~~ :-

Diagram :-



syntax :-
```
class superclass
{
  =
  =
}
class subclass1 extends superclass
{ =
  =
  =
}
class subclass2 extends subclass1
{ =
  =
}
```

Display add, division, multiplication subs in a single java program using inheritance property.

```
class A
{
int a, b, c;
```

```
void add()
{
a=10; b=20;
c=a+b;
system.out.println(c);
}
}
class B extends A
{
void sub()
{
a=20; b=10;
c=a-b;
system.out.println(c);
}
}
class C extends B
{
void mult()
{
a=2; b=4;
c=a*b;
system.out.println(c);
}
}
class D extends C
{
void div()
{
~~int~~ a=4; b=2;
c=a/b;
```

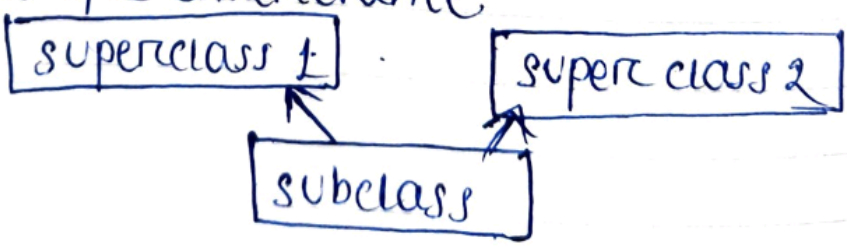
```
system.out.println(c);
    }
class Result
    {
public static void main(String args[])
    {
  D abcd = new D();
    abcd.add()
    abcd.sub()
    abcd.Mult()
    abcd.div()
        }}
```

• O/P - ? c = 30
c = 10
c = 8
c = 2

(3) Multiple Inheritance :-

When one subclass ~~once~~ wants to inherit the property of two or more superclass. That method is called as multiple inheritance.

Diagram :-



duplication)

Why Java doesn't support multiple inheritance (memory

Whenever a subclass wants to inherit the property of two or more superclasses that have same method, at that time java compiler cannot decide which class method it should inherit. Then there might be some chances of memory duplication that is a reason java doesnt support multiple inheritance through classes.

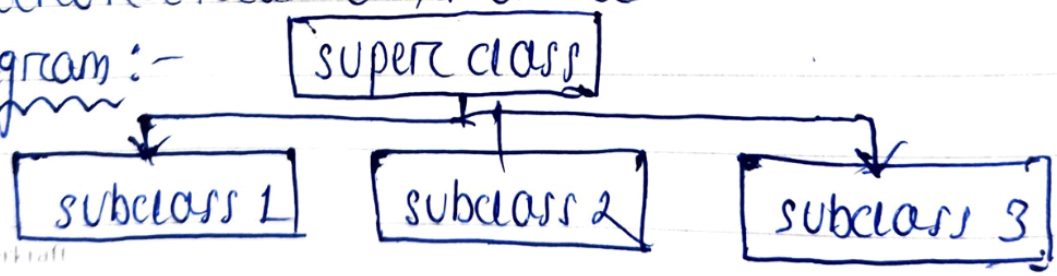
syntax:

```
class A
  { statement;
  }
class B
  {
   statement;
  }
class C extends A, B
  { statement;
  }
```

(4) Hierarchical inheritance :-

An inheritance which contains only one superclass and multiple subclasses and all subclass directly extends from its superclass called Hierarchical inheritance.

Diagram :-



syntax :-

```
class superclass
  {
  }
class subclass extends superclass
  {
  }
class subclass2 extends superclass
  {
  }
```

program :-

```
class A
  {
int a, b, c;
void display ()
  {
system.out.println ("Result is 2 :");
  }}
class B extends A
{ void add ()
  {
a=10; b=20;
c=a+b;
system.out.println (c);
  }}
```

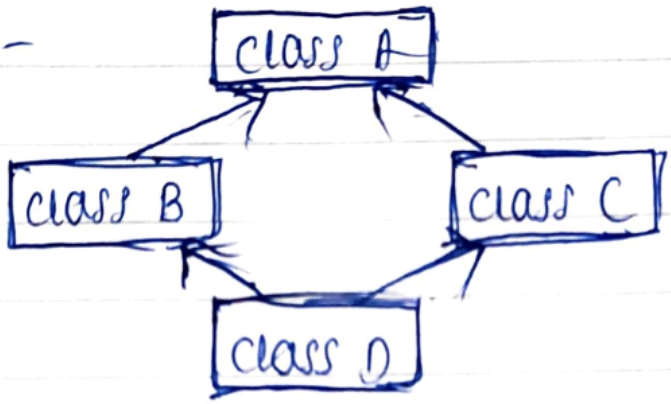
```
class C extends A
{
void sub
{
a=10; b=5;
c=a-b;
system.out.println(c);
}}
class Test
{
public static void main(String args[])
{
B ab=new B();
C ac=new C();
ab.display();
ab.add();
ac.display();
ac.sub();
}}
```

output = Result is
30
Result is
5

(5) Hybrid Inheritance :-

It is the combination of more than one type of inheritance is called hybrid inheritance. It is also not be supported by Java due to the presence of multiple inheritance.

Diagram :-



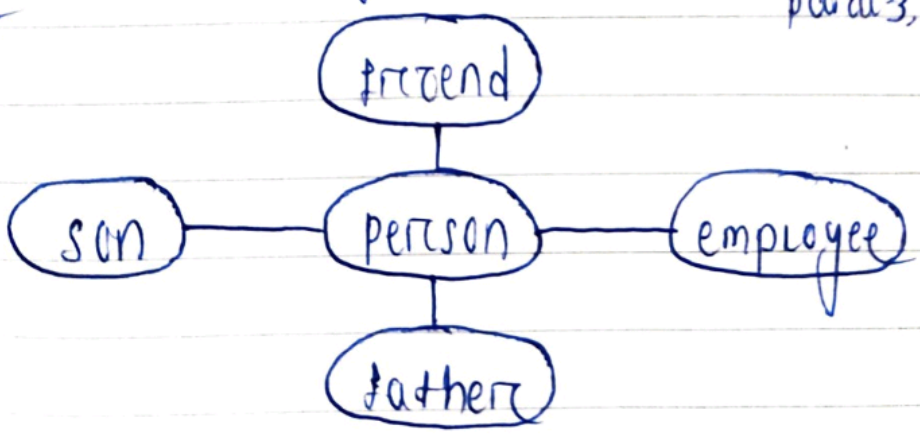
POLYMORPHISM :-

Date :- 11-12-2023-Monday

polymorphism is a Greek word whose meaning is same object having different behaviour.

syntax :- Returntype method-name ( )
Returntype method name (parameter 1)
Returntype method name (parameter 1, parameter 2, para 3, 4 - - - - - -))

Ex :-



void person (friend)
void person (employee)
void person (father)
void person (son)

polymorphism is of two types:-

① compile time polymorphism
② Runtime polymorphism

① **compile time polymorphism:-**

A polymorphism which exists at the time of compilation is called compile time or Early binding or static polymorphism.

Ex:- Method overloading

Method overloading:-

Whenever a class contains more than one method with same name and different types of parameters is called method overloading.

syntax:- Returntype Method name ( )
Returntype method name (parameter 1)
Returntype method name (parameter 1, parameter 2, para 3, 4 - - - - -)

```
class A
{
int a, b, c;
void add ()
{
```

```
        a=10;
        b=20;
        c=a+b;
system.out.println(c);
        }
    void add(int x, int y)
        {
        c=x+y;
system.out.println(c);
        }
    void add(int x, double y)
        {
        double c;
        c=x+y;
system.out.println(c);
        }
    void add(double x, double y)
        {
        double c;
        c=x+y;
system.out.println(c);
        }
public static void main(string args[])
        {
```

Paperkraft

```
A r = new A ();
    r.add(3, 2.5);
    r.add();
    r.add(5,6);
    r.add(5,2.5);
    r.add(2,3,4,2);
        }}
```

Runtime polymorphism:-

A polymorphism which exists at the time of execution of the program is called Runtime polymorphism.

Ex:- Method overriding

Here inheritance must be required.

syntax:-
```
class super
    {
    void show()
    }
class sub extends super
    {
    void show()
    }
```

(A new type of object creation is taken here)

```
super s = new sub()
```

(when we can try to remove method overriding a key word that is :- super ();)

Method :-

Rule-1: First JVM checks the superclass method. If the method is not present in superclass then compilation error will occur here.

Rule-2 :- If the method is present in the super class then it will check the method is over ridden or not. If the method is over ridden method then refers subclass method. If the method is not over ridden then it calleds super class method.

Ex :-

```
class A
{
void display()
{
system.out.println("Hello");
}}
class B extends A
{
void display()
{
system.out.println("Hi");
}}
```

```
class Result
{
public static void main (String args[])
{
A r = new B();
r.display();
}}
```

(condition-1) O/P = Hi
(condition-2) O/P = compilation error
(condition-3) O/P = Hello
(condition-4) O/P = Hello (super.display())
Hi

Advantages :-

Although method name behaviours all are same then it gives the different output.

Package :-

Date :- 12-12-2023-Tuesday

A package arrange the number of classes interfaces and sub packages of same type into a particular group.

It is nothing only create some folder inside the windows.

Types of package :-

There are two types of package :- ① predefined pack

① pre-defined package :-

The package which is defined by java software is known as pre-defined package

① Java.Lang

Language supported package is a default package in which it suppose the language of java it is also known as heart of the java language.

② Java.util

All the classes and interfaces are present inside the java.util

③ Java.io

File handling input and output application present in java.io package.

④ Java.Applet

⑤ Java.awt

⑥ Java.net

All these three packages are used for the internet uses and web page design.

⑦ Java.sql

It is used for data base linked with java then is access modifier.

Access modifier :-

How all the data members are accessed in which situation generally there are four types of access modifier :-

① private

It is accessed within the class but not access by within the package outside package by subclass outside package.

② Default:

Accessed within the class and package but not accessed by outside package through subclass and outside package.

③ protected:-

Accessed by within the class ~~with~~ package outside package by/through subclass but not outside package

④ public

It is accessed by within the class within the package outside of the package ~~and~~ by subclasses and out side package

Advantages:-

(i) Reusability

(ii) security

(iii) Avoid naming conflicting

(iv) Fast searching

(v) Hiding can be used by using encapsulation.

Disadvantages:-

We cannot pass parameter to the package.

New chapter

Exception Handling :-

Exception :-

An exception is an unexpected, unwanted and abnormal situation that occur at runtime.

Exception Handling :-

In exception handling we should have an alternate source through which we can handle the exception then how to handle the exception or techniques for exception handling.

The object orientation mechanism has provided the following techniques to work with exception.

1. try ~~catch~~
2. catch
3. through
4. throws
5. Finally

If the exception is predefined then try and catch exception will be used if there is user defined exception then we use through or throws expression

If the user wants to execute the program wheather the exception is present or not then finally exception will be executed.

Throwable is the root class of java exception

heirarchy which has two subclasses that is exception and error.

Exception :-

An event which occurs during the execution of a program that disturbs the normal flow of program.

(a) Runtime exception
(a) Arithmatic exception
(b) NULL pointer exception
(c) Number formating exception
(d) Index out of bound exception

Array index out of bound Exception

string index out of bound exception

IO Exception :- (i) End of file exception
(ii) File not found exception

Error :-

An error describes any issue that arises unexpectly that causes a computer not function properly. It may be software or hardware error :-

(i) Stack of flow error
(ii) out of memory error
(iii) Input / output error
(iv) Linkage error

try and catch :-

try :- whenever we write a statement and if the statement is error suspecting statement or risky code then put that code inside the tryblock.

catch :- The main purpose of catch block is to handle the exception which are throws by tryblock.

catch block will not be executed if there is no exception inside the tryblock.

```
class A
{
public static void main (string args[])
{
int a, b, c;
a = 5;
b = 0;
c = a/b;
system.out.println(c);
}
}
```

```
try
{
c = a/b;
system.out.println(c);
}
catch
```

(In this program the part inside try and catch is not execute) but another part is execute.

```
try {
    c = a/b;
    system.out.println(c);
    }
    catch (AE e)
    {
s.o.p ("Arithmatic expression error")
```

If error is found in try block then catch part is execute, else error is not found in try block then catch block is not execute.

Types of Exceptions:- Date:- 13-12-2023 - Wednesday

1. Arithmatic Exception
2. Number format Exception
3. Null pointer Exception
4. String Index out of Bound Exception
5. Array Index out of Bound Exception

(int h = Integer.parseInt(d))

(String W = e.toUpperCase();)

Program:

```
class A
{
public static void main(String args[])
{
int a=5, b=0, c;
String d = "abcd"; String h = "1,2,3,4";
String e = null; String x = "synergy";
int a[] = {10, 20, 30, 40}
```

Paperkraft

```
try {
 c = a/b ;
system.out.println ( c );
        }
  catch ( Arithmatic expression a1)
        {
  system. out. println ("Arithmatic .Exception");
try{ int l = Integer . parseInt (h);
   system. out. println (l);
   }
   catch (Number format Exception . a2)
     { system. out. println ("Number format Exception");
       }
try { int k = Integer . parseInt (d)
   system. out. println (k);
   }
   catch (Number format Exception . a3)
   { system. out. println (" Number format exception");
try {
   int w = e. to upper case ();
   system. out. println (w);
   }
   catch (Null pointer Exception a4)
     {
     system. out. println ("Null pointer Exception")
       }
```

Multiple Exception :-

```
try
{
  statement 1
  statement 2
  statement 3
  }
catch (A.E.e)
  {
  statement 4
  }
catch (NPE n)
  {
  statement 5
  }
catch (ARROOBE i)
  {
  statement 6
  }
catch (Exception.m) → (super Exception)
  {
  statement 7
  }
```

Finally Block Exception:-

Finally Block is a real time block which is use to handle the resources like, security related code and database related code.

Each time it will execute wheather the exception is present or not.

We can use it by the chain of try-catch-finally.

Ex:- syntax:- main():

```
try {
{statement 1
 statement 2
 statement 3
    }
catch (Arithmetic Exception e)
    {
system.out.println("statement 4")
    }
Finally
    {
system.out.println("statement 5")
    }
statement 6
```

condition:- When statement-1 is correct, statement-2 is a Arithmatic Exception and statement 3 has Null pointer exception

st 1 ✓
st 2 A.E. Exception
st 3 N.P Exception

case-1

① then executing statements are - 1, 4, 5, 6 (T) (T means terminal-6)
(Because catch is ~~be~~ successfully handle the Arithmetic Exception which is found in st 3)

case-2

① When statement 2 is have a null pointer exception then

executing statements are - 1, 5

(In this case statement-4 and statement-6 is not execute because catch can't found the Exception)

Program :-

```
class ~~Formthy~~ Finally
{
public static void main (string args[])
{
int a=5, b=0, c, d;
try{
    c=b/a;
    system.out.println (c);
    d= a/b;
    system.out.println (d);
    }
catch (Arithmetic Exception a1)
    {
    system.out.println("Exception");
    }
Finally.
    {
    system.out.println ("Not Found"); }
    system.out.println ("Hi");
        }
}
```

Throw keyword :-

Throw key word is used to throw the user defined or customized exception object to the JVM explicitely for that purpose we use throw keyword.

Ex : ① main()
{
throw new InvalidRiseException
("sum can't rise in west");
{
system.out.println(" InvalidRiseException");
}

② main()
{
throw new ~~InvalidRiseExcept con~~ ArithmeticException
("/0 rise ArithmeticException")
}

Throws Exception :-

Throws key word is reuse when user doesn't ~~found~~ want to handle the exception to the JVM.

program:- import java.util.*;
import java.io.FileInputstream;
import java.io.FileOutputstream;
class RW
{

```
fileread ( ) throws FileNotFoundException
  {
   FileInputstream fs = new FileInputstream (E:/
                                              b.txt)
  }
filesave ( ) throws FileNotFoundException
  {
FileOutputstream fo = new FileOutputstream (E:/C.txt)
  }
  class A
   {
public static void main (string args[])
   {
  RW r = new RW ( );
   {
   try
    {
     r.fileread ( );
   system.out.println ("File found");
        }
   catch (FileNotFound Exception F)
        {
system.out.println ("File Not Found");
        }
        }
```

Throw	Throws
(i) Throw key word is used to throw an exception object explicitely.	(i) Throws key word is used to declare an exception over the method and pass the colour
(ii) Throw key word is always present inside the method body.	(ii) Throws key word always used the method explicately.
(iii) We can throw only one exception at a time.	(iii) We can handle multiple exception using throws exception.