- Interface -

Interface is same as ~~to~~ abstract class but its some differences from abstract.

In Interface are already all ~~classes~~ methods are abstract ~~already~~ (therefore we can't implement any method interface, but we use default access modifier we can implement the methods.)

~~In Interface we can use interface~~
In Interface we can use implements key word for ~~us~~ implementing the Interface.

In abstract we can use extends keyword for ~~ex~~ extends the ~~in~~ abstract class

~~In Interface the name~~
In Interface we can't change the attributes of Interface, because we only ~~implement~~ implements the ~~therefore~~ we can only access ~~to~~ the properties not be change, but in abstract class we are extends the class therefore we can change/update the properties

Page No.
Date

Example:

Abstract	Interface
abstract class Car {	Interface Car {
int a = 500;	int a = 500;
abstract void brake();	void brake (); }
create abstract	already abstract
}	
class Maruti extends Car {	class Maruti implements Car {
void brake () {	public void brake () {
sout ("Stop"); }	sout ("Stop"); }
	use public in interface
public class main {	public class main {
psvm (String[] args) {	psvm (String[] args) {
Maruti ev = new Maruti();	Maruti ev = new Maruti
ev. brake ();	ev. brake ();
ev. a = 3;	~~ev. a = 3~~
Sout (ev. a);	sout (ev. a);
update the abstract class data	we can't be change because a = 500 it is a final variable
Output:	Output:
brake 3	brake 500

Java ~~Supp~~ not Support multiple inheriten but with the use of Interface support multiple Inheriten -ce

abstract Vs Interface

~~Ex interface~~

In Interface we can implements a multiple classes in one class.

but in abstract we can only imple-ment a one class

Interface ki wajha se hum bohot sari class se ak single class bna sakte or class ko implement karne ke sath s-sath extend bhi kar sakte h. without Inter-face ye possible hi nahi h ki ki hum multiclasses se ak single-class bna sakte.

Ex.

```
Interface cat {
    void flexible () ; }

Interface dog {
    void powerful () ; }

class Animal implements cat, dog {
    void flexible () {
        sout ("flexible") ; }
    void powerful () {
        sout ("powerful") ; }
```

implement

Inherit multiple classes in a single class

Splender Sp = new Splender
Reference object

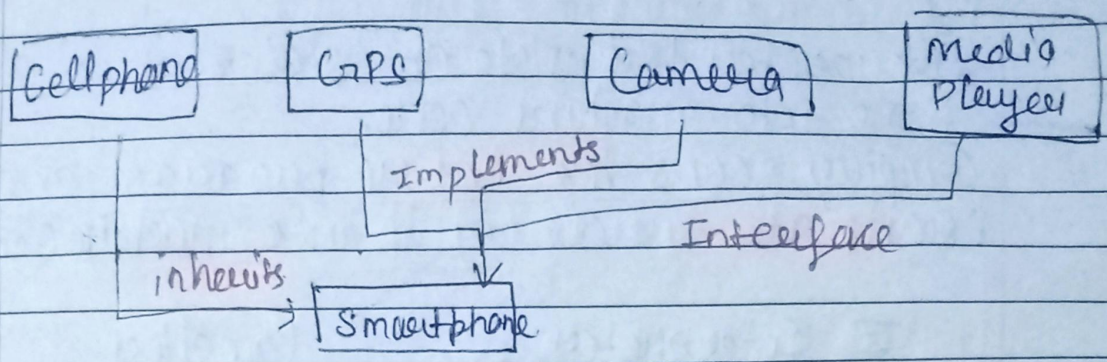
Interface Defaults Methods

~~Interface~~ If we add the default access modifier in Interface method, then this method not be abstracted.

Ex.

```
Interface Bike {
    void brake ();
    default void Diskbrake () {
        sout ("add diskbrake"); } }
class Splendor implements Bike {
    void brake () {
        Sout ("apply brake"); } }
class main {
    psvm (String[] args) {
        Splendor Sp = new Splendor();
        sp.brake ();
        sp.diskbrake ();
```

Polymorphism in Interfaces



GPS g = new Smartphone (); can only use GPS methods

Smartphones s = new Smartphone (); use all smart-phones methods

entire → all

Page No.
Date

INTERPRETER V/S Compiler

Interpreter translate one statement at a time into machine code.
Compiler scans the entire program and translate whole of it into machine code.

Interpreter	Compiler
• One statement at a time	• Entire program at a time
• Interpreter is needed every time	• Once compiled it is not needed
• Partial Execution if error	• No execution if an error occurs
• Easy for programmers	• Usually not as easy as interpreted ones
• Slower than Compiler	• Faster

→ Is Java compiled or Interpreted?
Java is a hybrid language → both compiled as well as interpreted.

Java File (harry.java) —Compiled using javac→ Class File (harry.class) → Can be used by Java interpreter

↓

bytecode

→ A JVM can be used to Interpreter this bytecode.
→ This bytecode can be taken to any platform (Win/Mac/Linux) for execution
→ Java is a platform independence

Executing a Java program :-

Javac File-name.java → Compiled

Java Filename.class → Interpreted

Packages in Java :-

- Built in packages → Java API
- User defined package → custom package

Annotation

An annotation in java is like a label / tag that you can put on classes, methods, variable, to give them some extra information.

Example : @Override

Page No.
Date

Using a Java package

import java.lang.* → import everything from java.lang

```
import java.util.Scanner;

public class Main {
        psvm (String[] args) {
            Scanner sc = new Scanner(System.in);
                }}
```

Types of declaration

```
public class Main {
        psvm (String [] args) {
java.util.Scanner sc = new java.util.Scanner(
                System.in);
                }}
```

Access Modifiers

	With in class	With in package	outside Package	Outside Package by subclass only
Public	Y	Y	Y	Y
Protected	Y	Y	N	Y
Default	Y	Y	N	N
private	Y	N	N	N

Multithreading

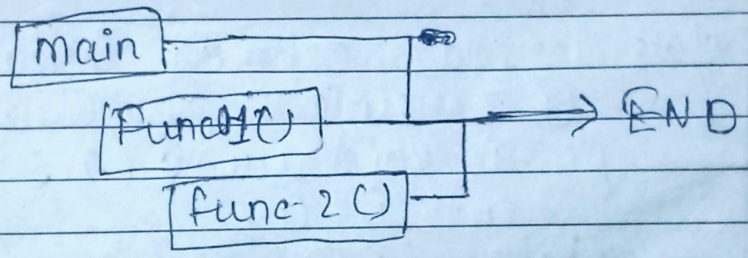
Multiprocessing and multithreading both are used to acheive multitasking.

Flow of control of Java

Without threading

main → func1() → func2() → END

With threading



Suppose hamare pass 2 class h x and y dono ke andar loops angar me bina thread ka use karte to jo me phele fn ko call karunga vo run pahale hoga par agar me threading ka use karu to dono sath-sath run honge.

Threading ka use karne ke liye hame class ke bad extend Threads key word ka use karna h.

extra:- aur hame import java.lang.*; ka bhi use karna hota h.

Page No.
Date

Example threading :-

```
import java.lang.* ;
class Thread1 extends Threads {
    @Override
    public void run() {
        int i = 0;
        while (i<10) {
            sout ("Thread 1st ") ;
            i++
            } } }
```

Override this method form thread class

API of Java help to create thread

```
class Thread2 extends Threads {
    @ Override
    public void run () {
        int i = 0;
        while (i<10) {
            sout ("Thread 2nd ") ;
            i++ } } }
```

without this method override .start() not worked

```
class Main {
    psvm (String[] args) {
        Thread1 t1 = new Thread1 ();
        Thread2 t2 = new Thread2 ()
        t1.start ();
        t2.start () ; } }
```

. start is Threads class fn run automatically method with use the name?

Output
Thread 1st
Thread 1st
Thread 2nd
Thread 2nd
Thread 1st
Thread 2nd

→ In Multi-threading there are two ways to used this.

- using Inheritance (extends Thread)
- using Interface (Implements Runnable)

Thread class extend karte h to jarurri nahi h ki hume run() method ko override karna compulsory h. but agar hum Runnable implements ka use karte h to run() method ko use karna compulsory h kyoki interface method abstract hote h.

Page No.
Date

Example of Runnable :-

Interface

```
import java.lang.*;
class Thread1 implements Runnable {

    public void run() {
        while (True) {
            sout (" Thread 1 run "); } } }

class Thread2 implents Runnable {
    public void run() {
        while (True) {
            sout (" Thread2 run "); } } }

class Main {
    psvm (String[] args) {

        Thread1 bullet1 = new Thread1();
        Thread gun1 = new thread (bullet1);

        thread2 bullet2 = new thread2();
        Thread gun2 = new thread (bullet2);

        gun1.start();
        gun2.start(); } }
```

abstract method override is compulsory

phale hum class ka object banate h phir hum Thread feature ka use karke new object banate h phir usse purane object se relate karte h.

LIFE CYCLE OF THREAD

NEW
↓
Runnable
↓
Terminated

NEW
↓
BLOCKED (non-Runnable)
↓
Terminated

(i) New → Instance of thread created but not started by invoking .start().

(ii) Runnable → After invocation of .start() thread in Running.

(iii) Blocked → Thread alive, not eligible to Run. Ex – agar me thread infinite loop lga du aur main fun me bhi ak infinite loop lga du jo thread ki state ko print karta ho. to jab main ka infinite loop Chalega to thread ka loop nahi chalega to hence thread ki state BLOCKED show hogi.

(iv) Terminate – then we exit from run() method.

Use .getPriority for set priority

.getPriority () — Used for Print Priority

→ How to set thread priority ?

- Minimum priority → 1
  Mid priority → 5
  Max priority → 10

Har thread me default Mid priority set hoti h. (In thread priority is already set 5 is default priority :)

~~Use .get Pri~~

Example of set priority

```
import java.lang.Thread;

class Thread1 extends Thread {
    pubic void run () {
→ while(true) {System.out("This is first thread"); }}}

class thread2 extends Thread {
    public void run () {
→ while(true) { System.out("It is Second Thread"); }}}

public class Main {
    psvm (String[] args) {
        thread 1 t1 = new thread1();
        thread 2 t2 = new thread2();
        t1.getPriority (8); → set priority
        t1. start () ; t2 .start();
```

both in infinite loop

t1 priority 8 and t2 priority 1

but t1 run more than to t2 because t1 priority is higher than t2.

~~Set priority~~
How to set priority in Thread1 class
class

```
import java.lang.Thread;
class Thread1 extends Threads {
    public void run() {
        while (true) {
            Thread.currentThread().getPriority(9);
            sout("This is 1st Thread");
}}}
```

import class — used for current thread — set priority function

- ~~get~~ .getId() and .getName()

sout (Thread.getId()); → for find the id of thread.

sout (Thread.getName()); → for find and give name of thread. use of constructor give the name of Thread.

Ex.; ~~class thread1 extends Thread {~~
construction ← thread1 (String name) {
call to thread class ← super (name);
void run() { sout ("thread 1") } }

used super

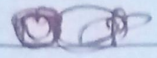
Page No.
Date

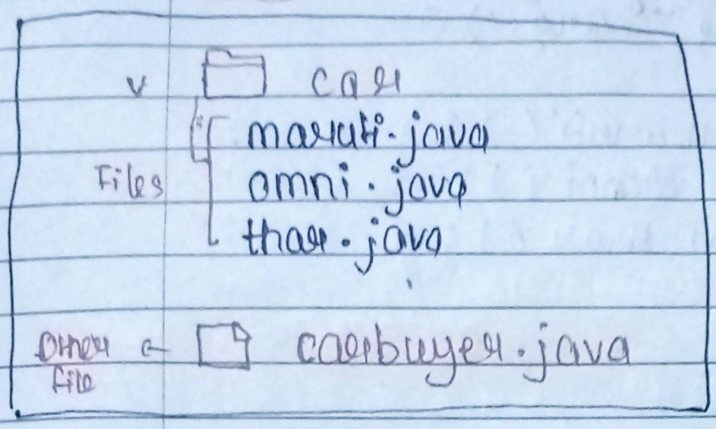
.join() → ~~ye~~ ~~[illegible]~~ main to agar hame pehale t1 ko pura complete run karna h phir uske bad hame t2 ko run karna to .join() hame ~~please~~ thread ko ek sath run karne me help karta h.

~~Thread~~ .sleep(time in millisecond) - It used for ~~to~~ create delay in program.

— PACKAGE —

- Create a custum package ;





we can use a of psvm (strings) for creating a package.

maruti.java

```
→ package.car;
  public class maruti {
      public void speed () {
          sout ("car name is maruti"); }}
```

thar.java

```
→ package.car;
  public class thar {
      public void speed () {
          sout ("car name is thar"); }}
```

Omni.java

```
Package.car
  public class Omni {
      public void speed () {
          sout ("car name is omni") ; }}
```

Page No.
Date

carbuyer.java

```
import car.*;
```

Import package
* mean with all class.

```
public class carbuyer {
    psvm (String[] args) {
        maruti m = new maruti();
        Omni o = new Omni();
        thar t = new thar();
        m.speed();
        o.speed();
        t.speed();
    }
}
```

but in we use psvm for access package

create Object

Output :

car name is maruti
" " " omni
" " " thar

Errors & Exceptions

There are three types of Errors :-
(i) Syntax error
(ii) Logical error
(iii) Runtime error → Also called Exception!

→ Syntax error

A syntax error is a type of error that occurs when the syntax of a programming language is violated. In simpler terms, it means that the code does not follow the rules programming language grammar.

Ex →
→ Missing semicolons, parentheses, brackets.
→ Misspeled keywords
→ Using an Operator incorrectly

Misspeled mean multiple meanings

→ Logical Error

Logical error occur when a program compiles and runs but does the wrong thing.

Page No.
Date

Example :-

```
public class LogicalerrorEx {

psvm (String [] args) {

    int num1 = 5 ;
    int num2 = 3 ;

    int result = num1 / num2 ;

    Sout ( result ) ; } }
```

It is a logical error. because output = 0 we divide larger no. to smaller no.

→ Runtime Error (Exception)

Java may sometimes encounter an error while program is running. These are also called Exception.

for example :→ Suppose that you create a 2 digit Numbers add program. you are take input to user of both no. User 1 no. input is 5 and second no. input is 'a'. In this you program thrown an a run-time error or expection. because int not be store character value.

Note :→

Syntax and logical error are encountered by the programmer but Runtime error are encountered by the users.

Page No.
Date

EXCEPTION

Exception is an unwanted or unexpec-ted event, which occurs during the execution of a program, at a run time, that ~~dist~~ disrupts the normal flow of the program's instruction.

There are mainly two types of exceptions in java :-

1. Checked Exception → Compile time exception (Handle by compiler)
2. Unchecked Exception → Runtime exceptions

1. Checked Exception - In this Exception compile checked the program at compile time give the warning to user for handle the ~~exper~~ exception.

Example of checked Exception :-

→ Thread.sleep (int)
→ Thread.join ()

Then we use the this function sleep and join then compile shows then error before run therefore we use the try/catch block for Handling the exception.

2. Unchecked exception → Unchecked exception occurs in runtime.

for example: You are a program take two input from user and divide the second input to the first input, In your first Input is 50 and second is 0

Then you show that JDE throw an Arithmetic exception because we divide any no. to 0.

Page No.
Date

~~Exep~~ Exception Handling

Try-catch is the simplest method of of handling the Exceptions.

Try-catch block

In java, ~~exep~~ exceptions are managed using try-catch blocks.

Syntax :-

```
try {
    // code to try
}
catch (Exception e) {
    // code if exception }
```

Ex :-

```
class main ( ) {
    psvm (String args[]) {
        j.u.Scanner sc = new j.u.Scanner (system.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        try {
            c = a/b ;
            sout("The result is" + c); }
        catch (Exception e) {
            sout (" We failed for some reason"); }
```

Input from user

In this program than occurs the any exception the we catch the exception and run this code

Handling specific Exception

In java, we can handle specific exceptions by typing multiple catch blocks.

```
try {
    //code }
catch (ArithmeticException e) {
    // code }
catch (ArrayIndexOutOfBoundsException e) {
    // code }
catch (Exception e) {
    // code }
```

for handle Arithmetic exception

for handle array index exception

for all other exceptions

Nested try-catch

Syntax :-

```
try {
    // code
    try {
        //code }
    catch (Exception e) {
        //code }
    }
catch (Exception e) {
    // code }
```

We can use further nest try catch blocks inside the nested try catch blocks.

Page No.
Date

Example : Nested try catch

```
Import java.util.Scanner;

class Nested_try_catch {

  psvm (String [] args) {

    int a, b, c, d;

    Scanner sc = new Scanner (System.in);

    a = sc.nextInt ();
    b = sc.nextInt ();

    try {

      c = a/b;
      sout (c);
      try {

        d = sc.nextInt ();
        d = c/d; }
      catch (Exception e) {
        sout ("Level 1 Exception");
    } catch (Exception e) {
      sout ("Level 2 Exception"); }
```

If you enter the value of b=0 then runs the Level 2 Exception

You enter the value of d=0 then catch the level 1 exception

EXCEPTION CLASS

Create custom exception

We can create our own Exception using Exception class in Java.

Syntax :-

```
public class MyException extends Exception {
    // Code
}
```

The exception class has following important methods :-

1. String toString () → executed when sout(e) is ran
2. void printStackTrace () → prints stack trace
3. String getMessage () → print the exception message

And used your methods →

to

Page No.
Date

Create custom exception :-

```
import java.util.*;

class myexception extends Exception {

    public my exception () {
        sout("This method run auto without call.") ; }

    public void exp2 () {
        sout("This is custom exception"); }

}

class customexception {
    psvm (String[] args) {

        scanner sc = new Scanner (System.in);

        int a = sc.nextInt();

            if (a==0) {
                throw new myexception; }

        catch (myexception e) {
            e.exp2 ();
            sout("print other thing"); }

}}
```

Cons tructor

method

Input

throw an exception

call exception method by add e.method

THROW & THROWS

- 'throw' is used to throw an exception with in a method.

'throws' is for declaring which exceptions a method might throw.

In easy language throw used throw exception in method can throw used declaring a exception.

Example :-> Throws

```
public class Main {
    ~~psvm (String[] args) {~~

    ~~static void UseThrowand~~
    static void UseThrowandThrows (int a, int b)
        throws ArithmeticException {
            ~~if (b == 0)~~
            int c = a/b; }

    psvm ( String[] args ) {
        int a = 5;
        int b = 0;
    try { UseThrowandThrows (a, b); }
    catch ( ArithmeticExpression e) {
        sout ("Throws work properly"); } }
```

Create method Throws Arithmetic expression

Use try-catch because this method throw Arithmetic Expression.

Page No.
Date

Example - Throw

```
public class Main {
    psvm (string [] args) {
        int a = 5;
        int b = 0;

        if ( b == 0 ) {
            throws new ArithmeticExpression(" Enter
            your message ") ; }
        else {
            int c = a/b ;
            sout (c) ; }
    }}
```

Output :- Enter your message