Introduction Of Java

Java language सन माइक्रोसिस्टम का प्रोडक्ट है यह language सन 1991 में जेम्स गोसलिंग, पैट्रिक नॉटन ,क्रिस वोथ ,इडी फ्रेक एवं माइक शेरिडन के द्वारा विकसित की गई थी प्रारम्भ में यह language oak के रूप में जानी जाती है परन्तु बाद में सन 1995 में इसका नाम Java पड़ा जावा एक ऑब्जेक्ट ओरिएंटेड language है अर्थात प्रत्येक ऑब्जेक्ट क्लास में उपस्थित होता है जावा का मेन function भी क्लास में लिखा गया है जावा प्रोग्राम क्लास ऑब्जेक्ट एवं मेथेडस के द्वारा ओरिएंटेड होता है जावा एक प्लेटफोर्म इंडीपेंडेंट language है अर्थात प्लेटफोर्म को एक ऑपरेटिंग सिस्टम में विकसित एवं दुसरे ऑपरेटिंग सिस्टम मे रन किया जा सकता है प्लेटफोर्म इंडीपेंडेंट बाईट कोड के द्वारा प्रदान की जाती है बाईट कोड एक कॉमन कोड है जिसे जावा कम्पाइलर द्वारा उत्पन्न किया जाता है और होस्ट के एक्जीक्युटेबल कोड के अनुसार किसी भी ऑपरेटिंग सिस्टम के द्वारा इनरप्रिटर किया जा सकता है इस language की एक विशेषता यह है की जिस मशीन पर बाईट कोड रन होता है उस पर जावा वर्चुअल मशीन (JVM) का उपस्थित होना आवश्यक होता है जावा इनरप्रिटर होती है जो की जावा बाईट कोड का उपयोग करने के साथ-साथ होस्ट के ऑपरेटिंग सिस्टम के इंस्ट्रक्शन कोड का उपयोग करके उसे एक्जीक्युटेबल कोड में परिवर्तित कर देती है JVM के सहयोग के बिना जावा प्रोग्राम एक्जीक्युट नही हो सकता है जावा प्रोग्राम को स्टोर एवं एक्जीक्युट करने की लिए एक्जीक्युटिव फ़ाइल बनाने की बजाए इसे एक्जीक्युटेबल कोड में बदलती है और JVM के द्वारा रन करती है

Fichars of Java

1. Object Oriented :

object Oriented का मतलब यह है की एक एसी तकनीक जिसमे ऑब्जेक्ट पर फोकस किया जाए object Oriented तकनीक कहलाती है object Oriented

तकनीक में हर चीज एक ऑब्जेक्ट होती है java एक object Oriented programming language है जिसके java c++ की तरह ही object Oriented language है किन्तु जावा के कुछ फीचर java को c++ से अलग बनाते है जेसे multipal inheritance

2. Distributed:

Distributed का तात्पर्य यह है की जावा प्रोग्राम को नेटवर्क पर चलाया जा सकता है java के रिमोट प्रोग्राम को लोकल मशीन पर रन किया जा सकता है

3. Robust :

java एक Robust language है इसका तात्पर्य है की यह जावा में एक्सेप्शन हेंडलिंग का कांसेप्ट है जो एरर को cepchar कर लेता है और system क्रेशिंग की Risk को कम कर देती है

4. Secure :

java सिक्योर language है इसका मतलब यह है की जावा बहुत से सिक्योरिटी प्रदान करती है जेसा की हम जानते है की जावा के प्रोग्राम network पर भी चल सकते है जब कोई प्रोग्राम network पर चलता है तो उसे सिक्योरिटी की अवश्यकता होती है की कोई वाइरस उस प्रोग्राम को नष्ट ना कर दे java इस तरह के वाइरस से प्रोग्राम को सुरक्षित रखने का कार्य करती है इस लिए कहते है की जावा एक Secure language है

5. Compiled और Interpreted :

किसी भी प्रोग्रामिंग language में कम्पाइलर या इंटरप्रिटर दोनों में से एक जरुर ही होता है किन्तु जावा में ये दोनों है ये दोनों मिल कर जावा प्रोग्राम को

compile और Interpret करते है सबसे पहले कम्पाइलर जावा से सोर्स कोड को बाइट कोड में convert करता है उसके बाद इंटरप्रीटर बाइट कोड को मशीन कोड में convert करता है

6. Multithreaded :

Multithreaded को मतलब यह है की जावा में एक साथ कई process को एक साथ एक्सीक्यूट किया जा सकता है Multithreading कहलाती है

7. Platform Independent :

जावा एक Platform Independent language है इसका मतलब यह है की जावा के प्रोग्राम को एक कंप्यूटर पर compile कर के उसे किसी और कंप्यूटर पर उसे ले जा कर रन किया जा सकता है

java Environment in Hindi

java प्रोग्राम को रन करने के लिए जो Environment बनाया जाता है उसे java Environment कहा जाता है java Environment किट में क्लासेस और मेथड का एक set होता है जिसे JDK (java Devlopment kit) कहते है

java Devlopment Kit में निम्लिखित tool होते है जिनका use कर के java प्रोग्राम को compile और रन किया जाता है

1. javac :

इसका उपयोग जावा प्रोग्राम के सोर्से कोड को बाइट कोड में convert करने के लिए किया जाता है मतलब जावा प्रोग्राम को compile करने के लिए किया जाता है

2. java :

इसका उपयोग बाइट कोड को मशीन कोड में convert करने के लिए किया जाता है मतलब java प्रोग्राम को इंटरप्रीट कर के प्रोग्राम को रन करने के लिए इसका use किया जाता है

3. Appleteviewer :

इसका use Applet के प्रोग्राम को रन करने के लिए किया जाता है applete वह प्रोग्राम होते है जिनको वेब ब्राउसर पर रन किया जाता है

Basic structure of Java program

```
Import package;
Class classname
{
        Variable declaration ;
        Method

       main method
}
```

Example :

```
Import java.lang.*;      //package import
class Hello   //class creation
{
        Int i=10;   //variable Discretion
        void  Disp()   //method
        {
                System.out.println("Hello");
        }
        public static void main(String args[])  //main method
        {
              Hello obj=new Helllo();
```

```
            Obj.Disp();
        }
}
```

What Is Class

जेसा की हम जानते है की java एक object Oriented language है इस लिए इसमें क्लास का use किया जाता है class को एक container माना जा सकता है जिस तरह एक container में different – different type की चीजे रखते है use तरह जावा में class का use different – different type के variable और method को रखने के लिए करते है class कहलाती है class एक user define Data Type है

Syntax :

```
class class_name
{
    Body of Class
}
```

What Is Object In Hindi

जेसा की हम जानते है की जावा एक object oriented प्रोग्रामिंग language है इसका मतलब यह है की जावा में हर कार्य को एक ऑब्जेक्ट मान कर किया जाता है java में एक ऑब्जेक्ट किसी ना किसी class का reffrence होता है हर ऑब्जेक्ट किसी ना किसी class का एक reffrence होता है जिसका use कर के उस class के किसी भी मेम्बर को एक्सेस किया जा सकता है

Syntax :

class name reffrence variable = new Constrictor name(argument list);

Example :

Demo obj=new Demo();

What Is Method

Method Code का एक एसा ब्लाक है जिसके अन्दर उसकी स्वय की कोडिंग होती है मेथड का एक नाम होता है जिसके द्वारा उसे पहचाना जाता है एक मेथड को किसी ना किसी एक विशीष्ट कार्य करने के लिए बानाया है जावा में मेथड बिलकुल उसी तरह होती है जिस तरह c , c++ में function होते है मेथड कहलाती है

Syntax :

```
return_type method name(argument list)
{
        body of method ;
        return statement;
}
```

Example :

```
int add(int a,int y)
{
     int c;
    c=a+b;
```

```
    return c;
}
```

What Is JVM In Hindi

JVM का पूरा नाम जावा वर्चुअल मशीन है ये जावा प्रोग्राम को रन करने के लिए एक एन्वार्मनेट बनता है जिससे जावा का प्रोग्राम किसी भी प्लेटफोमर्‌ पर चल सके JVM का कार्य बाइट कोड को इंटरप्रीट कर उसे मशीन कोड में convert करना होता है जिससे प्रोगाम रन होता है JAM कहलाता है

Data Type in Java

java में जब भी कोई variable use किया जाता है तो उस variable का कोई ना कोई type होता है जिस type के कारण उसे memory में size allocate होती है जावा में अलग अलग प्रकार के datatype होते है जिनकी अलग अलग size होती है

java में Data type निम्लिखित है

1. Integer :

integer जावा का एक प्रीडीफ़ाईन Data type है जिसका use पूर्णक value को स्टोर करने के लिए करते है इसे int के द्वारा प्रदर्शित किया जाता है java में integer 4 बाइट का होते है

जेसे :

```
int x=120;
```

2. float :

float जावा का एक प्रीडिफाइन data type है इसका use डेसीमल value को स्टोर करने के लिए किया जाता है java में float 4 बाइट का होता है

जेसे :

float x=1.2;

3. Character Data type :

Java में करैक्टर एक प्रीडिफाइन data type है जिसका use जावा में करैक्टर को स्टोर करने के लिए किया जाता है जावा में करैक्टर को single कोड ‘ ’ के अन्दर लिखा जाता है

जेसे :

char latter =’A’;

4. Boolean data type :

जावा में बूलियन data type का use किया जाता है बूलियन data type में दो value होते है पहली true और दूसरी false

Variable in Java

variable एक एंटिटी है जो की एक स्टोरेज लोकेशन को प्रदर्शित करते है variable की value पर किसी भी प्रकार का कैलकुलेशन किया जा सकता है तथा प्रोग्राम के रन होने पर variable आवश्यकता पढने पर अपनी value चेज भी कर सकते है

Java में variable 3 प्रकार के होते है

1. Instance variable :

instance variable वो variable होते है दो class की अन्दर डिक्लेअर किया जाते है ये variable तब create होते है जब ऑब्जेक्ट बनाया जाता है अत: ये ऑब्जेक्ट के साथ जुड़े होते है ये प्रत्येक ऑब्जेक्ट के लिए एक अलग value प्रदर्शित करते है

2. Class variable

Class variable भी class के अन्दर ही डिक्लेअर किये जाते है class variable class में ग्लोबल होते है और class द्वारा बनाए गए सभी ऑब्जेक्ट से सम्बन्धित होता है प्रत्येक class variable के लिए केवल एक memory location बनाई जाती है और class के सभी variable उसे कॉमन फॉर्म में शेयर करते है

3. Local variable :

ऐसे variable जो किसी मेथड मे अन्दर डिक्लेअर किये जाते है लोकल variable कहलाते है इन्हें लोकल variable इस लिए कहा जाता है क्योकि इनको मेथड के अन्दर ही use किया जा सकता है इन variable को मेथड के बहार use नहीं किया जा सकता है

Data Encapsulation Java

Data Encapsulation का अर्थ डाटा मेम्बर और मेथड को एक सेल में बाइंड करना होता है जावा एक सुरक्षित सेल प्रदान करती है जिसमे data मेम्बर तथा मेथड उपस्थित रहते है Encapsulation के 2 प्रमुख उदेश्य होते है प्रथम यह डेटा और मेम्बर को ऑर्गेनाइज एलिमेंट के द्वारा इल्लीगल अस्सेस्मेंट से सुरक्षित करता है दूसरा Encapsulation यूजर द्वरा डिफाइन नए data type को भी

बनता है यह कोड की विभिन इकाई उत्पन करता है जो की कोड की अन्य इकइयो के द्वारा पुनः उपयोग में लाई जाती है जावा में क्लासेस Encapsulation को एक्सीक्यूट करती है

.

How to access class Member

जावा में किसी भी class के मेम्बर को एक्सेस करने के लिए सबसे पहले उस class का ऑब्जेक्ट बनाना होता है ऑब्जेक्ट बनाने के बाद उस ऑब्जेक्ट के reffrence variable के साथ class की मेथड या variable का नाम लिखा कर एक्सेस किया जा सकता है

Example :

```
class Add
{
    int a=10,b=5,c;
    void opp1()
    {
        c=a+b;
        System.out.println("Sum :"+c);
    }
}
class Example
{
    public static void main(String args[])
    {
        Add obj=new Add();
        obj.opp1();
    }
}
```

Command Line Argument

Command line argument command line से argument पास करने का एक तरीका है कोई भी इनफार्मेशन जब मेथड में pass की जाती है तब वह प्रेंथिसिस के अन्दर variable के set द्वारा रिसीव की जाती है ये variable पेरामीटर कहलाते है main Function की argument लिस्ट में argument

line से pass इनफार्मेशन की रिसीव करने की व्याख्या होती है इसका स्ट्रक्चर निम्लिखित है

```
cass Test
{
   public static void main(String args[])
    {
          //what to do
   }
}
```

माना उपरोक्त structure test.java नाम की फाइल मे स्टोर है इस structure में String args[] ,args Name के parameter की डिक्लेअर करते है जो की string class के इंडेक्स का एक ऐरे है string type का ऑब्जेक्ट करैक्टर string को स्टोर करता है इस स्तथी में args प्रोग्राम के Execution के दोरान उपस्थित कमांड लाइन आर्गुमेंट को रिसीव करते है जब यह प्रोग्राम कम्पाइल होता है तब कम्पाइलर Test.class नेम की फाइल बनाता है कमांड लाइन से इस प्रोग्राम की निम्लिखित प्रकार से रन किया जाता है

c:\>java Test ”One” ”Two”

यह जावा एक इंटरप्रिटर है जो प्रोग्राम को रन करता है जब प्रोग्राम रन होता है तो कमांड लाइन के 2 आर्गुमेंट पास होते है जो की args ऐरे द्वारा रिसीव किये जाते है इस Parameter को निम्लिखित प्रकार से स्टोर किया जाता है
args[0]=”One”
args[1]=”Two”

Token in java

एक जावा प्रोग्राम क्लासेस का समूह होता है और एक class Different - Different statement , method का समूह होती है किसी भी जावा में प्रोग्राम टोकन उस प्रोग्राम की सबसे छोटी इकाई होती है जेसे keyword , variable ,datatype , constent आदि जावा में टोकन होते है
java में टोकन निम्नलिखित प्रकार के होते है

1 . key-word
2. Identifiers
3. Operator
4. Separators

key-word in java

java language में सभी शब्द या तो कोई key –word होता है या तो Identifier key word का एक निश्चित अर्थ होता है जिसे यूजर द्वारा बदला नहीं जा सकता है इन्हें पूर्व निर्धारित Syntax के आलावा अन्य अर्थो में प्रोयोग नहीं किया जा सकता है इन्हें अंग्रेजी के small Latter में लिखा जाता है key – word को reserve word भी कहते है java language में 50 key word होते है

Identifiers in **java**

पहचान चिन्ह या Identifiers वेरिएबल समूह के अंतर्गत आते है इनका प्रयोग वेरिएबल ,Constent, function , Array आदि की पहचान के लिए किया जाता है जिनका use यूजर अपनी आवश्यकता के अनुसार करता है

Identifiers का नाम कुछ सामान्य नियमो के अंतर्गत निम्लिखित लिखित प्रकार से रखा जा सकता है –

- Identifiers का नाम किसी latter से शुरू होना चहिये
- key-word को Identifier नाम की तरह प्रयोग नहीं कर सकते है
- विशेष चिन्हों में Underscore का use कर सकते है
- English के सभी latter java language में use किये जा सकते है लेकिन c एक case Sensitive language है इस लिए c language में Capital और Small latter अलग – अलग मने जाएगे

Operator in **java**

Java में operator एक चिन्ह होते है जिसका use जो एक या एक से अधिक value लेता है और कोई रिजल्ट वापस करता है

java में निम्लिखित प्रकार के ऑपरेटर होते है

- Arithmetic Operator
- Relational Operator
- Logical Operator
- Assignment Operator
- Conditional Operator
- Bitwise Operator

Separators *in* **java**

Separators इंडीकेट करने के लिए use होते है की कोन सा statement कहा से शुरू हुआ है और कहा पर end हुआ है Separators कहलाते है
जावा में Separators निम्लिखित प्रकार के होते है

- Prenthisis ()
- bresess {}
- breket []
- Semicolon ;

- coma ,

Constrictor *in* java

Constrictor एक विशेष प्रकार की मेथड होती है लेकिन Constrictor जावा की सामन्य मेथड से थोडा अलग होता है Constrictor जावा की मेथड से किस तरह से अलग है use हम निम्लिखित प्रकार से समझेते है

- Constrictor का कोई भी return type नहीं होता है
- Constrictor का नाम वही होता है जो class का नाम होता है
- Constrictor को use करने के लिए उसे अलग से call करने की आवश्यकता नहीं होती है जब class का ऑब्जेक्ट क्रिएट होता है तो Constrictor call हो जाता है

  java मे Constrictor 2 प्रकार के होते है

  - Default Constrictor
  - Parameterized Constrictor :

1. Default Constrictor :

Default Constrictor एक एसा Constrictor होता है जो कोई argument नहीं लेता है यह Constrictor ऑप्शनल होता है यदि किसी class में Constrictor नहीं बनाया जाता है तो उस class के ऑब्जेक्ट के बनते समय कम्पाइलर उस class में एक default Constrictor बना देता है

Example

```
class Demo
{
   Demo()    // Constrictor create
        {
            System.out.println("Hello");
```

```
        }
}
class Example
{
    public static void main(String args[])
        {
            Demo obj=new Demo();   // Constrictor calling
        }
}
```

.

2. Perameterized Constrictor :

यह एक एसा Constrictor होता है जो कोई ना कोई argument रिसिव करता है Perameterized Constrictor कहलाता है

Example of Perameterized Constrictor :

```
class Demo
{
Demo(int a,int b)
{
int c=a+b;
System.out.println("Sum :"+c);
}
}
class Example
{
public static void main(String args[])
{
Demo obj=new Demo(5,4);
}
}
```

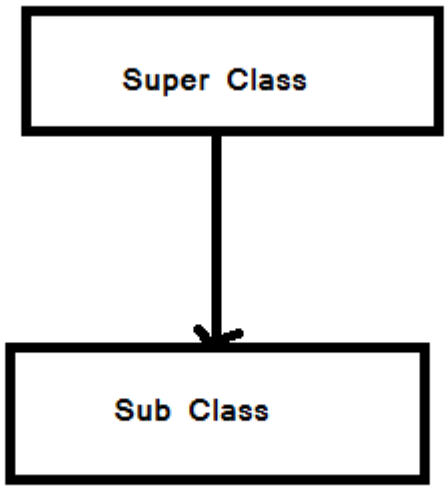
Inheritance In Java

java का एक महत्वपूर्ण गुण कोड रियुजेबलिटी है इसका अर्थ यह की जावा में यदि किसी कोड की एक ही प्रोग्रम में बार – बार जरुरत है तो उस कोड को बार बार लिखने की बजाए एक बार लिख कर उसे आवश्यकता अनुसार कभी भी use किया जा सकता है इसे कोड रियुजेबलिटी कहते है जावा में कोड रियुजेबलिटी Inheritance का use कर के की जाती है Inheritance जावा का एक कांसेप्ट है जिसमे जिसके दो या दो से जादा class का use किया जाता है जिसमे पहली class को super class तथा दूसरी क्लास को sub class कहा जाता है super class को sub class से Inheritance कर दिया जाता है अर्थात super class की सभी प्रोपर्टी sub class को दे दी जाती है Inheritance कहलाता है
java में Inheritance निम्लिखित प्रकार के होते है

- Single Inheritance
- Multilevel Inheritance
- Hierarchical Inheritance

1. Single Inheritance :

```
class A
{
   Body of class
}
class  B extends A
{
  Body of class
}
```



Single Inheritance

Program Of Single Inheritance

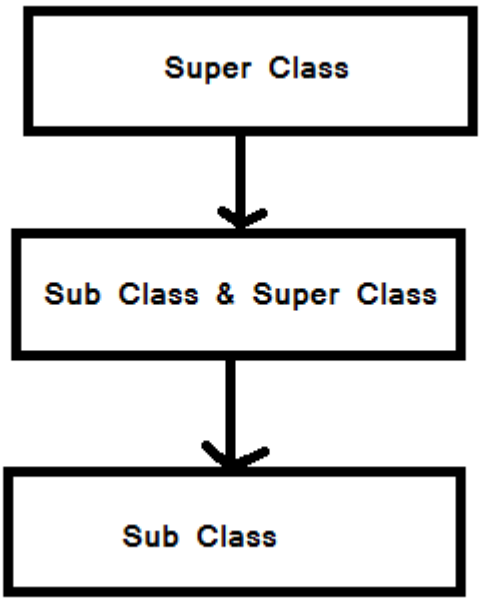
```
class A
{
  void m1()
  {
     System.out.println("Hello Class A ");
  }
}
class B extends A
{
  void m2()
  {
     System.out.println("Welcome Class B");
  }
  public static void main(String[] args)
  {
     B b=new B();
      b.m1();
     b.m2();
  }
}
```

2. Multilevel Inheritance:

जब किसी old class से new क्लास Derived की जाती है तो उस Inheritance को multilevel Inheritance कहते है

multilevel Inheritance का syntex निम्लिखित है

```
class A
{
  body of class
}
class B extend A
{
  body of class
}
        class C extends B
{
  body of class
}
```



Multilevel Inheritance

Program Of multilevel Inheritance

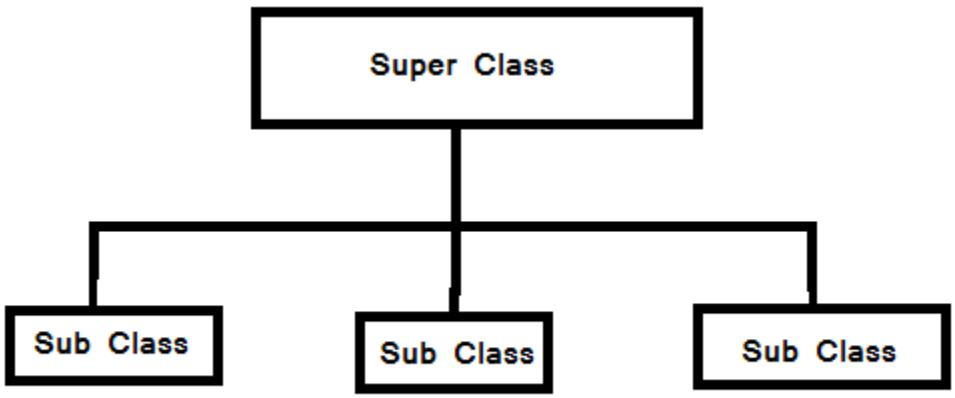
```
class A
{
  void m1()
  {
    System.out.println("Hello Class A ");
  }
}
class B extends A
{
  void m2()
  {
    System.out.println("Welcome Class B");
  }
}
class C extends B
{
  void m3()
  {
    System.out.println("Hello Welcome c");
  }
}
class D extends C
{
  void m4()
  {
    System.out.println("Class D");
  }
}
class Multi
{
  public static void main(String args[])
  {
    D d=new D();
    d.m4();
    d.m3();
    d.m2();
    d.m1();
  }
```

3. Hierarchical Inheritance :

इस Inheritance में एक super class की multipal sub class होती है Hierarchical Inheritance कहलाता है

Hierarchical Inheritance का syntex निम्लिखित है

```
class A
{
  body of class
}
class B extend A
{
  body of class
}
class C extends A
{
  body of class
}
```



Example Of Hierarchical Inheritance

```
class A
{
    void m1()
    {
        System.out.println("Hello A");
    }
}
class B extends A
```

```
{
    void m2()
    {
        System.out.println("Welcone B");
    }
}
class C extends A
{
    void m3()
    {
        System.out.println("Hello c");
    }
}
class D extends A
{
    void m4()
    {
        System.out.println("Hello D");
    }
}
class Herarical
{
    public static void main(String d[])
    {
        D d1=new D();
        d1.m1();
        d1.m4();

        C c=new C();
        c.m1();
        c.m3();

        B b=new B();
        b.m1();
        b.m2();
    }
}
```

Super Key Word In Java

C++ में sub class के Constrictor द्वारा super class के Constrictor को call किया जा सकता है यह करने के लिए c++ में sub class के Constrictor में पेरामीटर pass किये जाते है किन्तु जावा में इस तरह का कोई विकल्प है जावा में sub class से super class के Constrictor को call करने के लिए super मेथड का use किया जाता हो तथा super key word के द्वारा super class के सभी मेम्बर को एक्सेस किया जा सकता है

How To Use Super Class Constrictor Using super method
sub class में से super class के Constrictor को call किया जा सकता है sub class से super class के Constrictor को call करने का सिंटेक्स निम्लिखित है
super(parameter list);
java में sub class से super class के Constrictor को call किया जा सकता है किन्तु sub class से super class के Constrictor को call करने के लिए super मेथड sub class के Constrictor का पहला statement होना चाहिए

Example :

```
class A
{
	A(int x,int y)
	{
		System.out.println("Super Class :"+(x+y));
	}
}
class B extends A
{
	B()
	{
		  super(2,2);  //super class Constrictor call
		  System.out.println("Sub Class ");
	}
}
class Example
{
	public static void main(String[] args)
	{
		B obj=new B();
```

```
    }
}
```

final variable in java

final कीवर्ड का use constent को डिक्लेअर करने के लिए किया जाता है जावा में const reserve होता है इस लिए final का use किया जाता है final कीवर्ड के साथ डिक्लेअर किये गए variable को final variable कहते है जिसके लिए set value कभी नहीं बदलती है final variable हमेशा लोअर case में होते है final variable instence या class variable हो सकते है जब एक variable साधारणत : final कीवर्ड के साथ डिक्लेअर किया जाता है तब वह instence variable कहलाता है

final variable को निम्लिखित statement द्वारा समझा जा सकता है

```
class Finalclass
{
    final flaot RATE =12.5;
    static final double G=2344.2;
}
```

final Method in java

जावा का एक महत्वपूर्ण गुण मेथड ओवरराइडिंग होते है जिसमे super class के मेथड को पुनः डिफान किया जाता है किन्तु final मेथड का use से मेथड को ओवरराइडिंग होनी से रोका जा सकता है मेथड के साथ final कीवर्ड का use करने से मेथड ओवरराइडिंग नहीं होती है

```
class Emp
{
    final void disp()
    {
```

```
        System.out.println("Hello Good MoRMING");
    }
}
class Devloper extends Emp
{
    void disp()
    {
        System.out.println("Haw Are You");
    }
}
class Finaldemo
{
    public static void main(String a[])
    {
        Devloper x=new Devloper();
        x.disp();
    }
}
```

final class in java

जब कुछ class को इन्हेरिट होने से रोका जाता है तब उन्हें final कीवर्ड के साथ डिक्लेअर किया जाता है एक class को final डिक्लेअर करने से उस class के सभी variable और method final हो जाते है final class कहलाती है

Example :

```
final class Employe
{
    int s=1000;
}
class Devloper extends Employe
{
    void show()
    {
        System.out.println("Hello Goodmorning ");
    }
}
class Finaldemo
{
    public static void main(String d[])
```

```
    {
        Devloper x=new Devloper();
        x.show();
    }
}
```

Static Keyword in java

Java में class variable या class मेथड बनाने के लिए static कीवर्ड का use किया जाता है

जावा में जब भी किसी variable के नाम के पहले static key word लगा दिया जाता है तो वह variable class variable कहलाता है और उस variable को class की कोई भी मेथड use कर सकती है यह variable उस class के लिए एक global variable होता है

Method के नाम के पहले static कीवर्ड लगा देने से वह मेथड class मेथड बन जाती है तथा वह मेथड use क्लास के नाम के साथ इनवोक की जाती है class मेथड को class के नाम के साथ एक्सेस किया जाता है

Example :

```
class XYZ
{
   static int a=12;
   static int b;
   static void getSum()
   {
       System.out.println(a+b);
   }
   static
   {
       b=a+200;
       System.out.println("B :"+b);
   }
}
class SM
{
   public static void main(String[] args)
   {
        XYZ.getSum();
```

```
   }
}
```

Access Modifier in java

सभी public protected private Access Modifier कहते है जब ये सभी मोड Inheritance या package में use किये जाते है तो ये अलग अलग कार्य करते है package में ये access Modifier visibility control की तरह कार्य करते है जबकि Inheritance में ये एक्सेस के तरह कार्य करते है
Access Modifier निम्लिखित प्रकार के होते है

1. Public :

public Access Modifier instence variable ये मेथड पर लागु होता है तब ये variable या मेथड की sub class के ऑब्जेक्ट द्वारा डायरेक्ट एक्सेस की परमिशन देते है public Access Modifier अप्शनल होता है अर्थात कोई इसे applay नहीं करता है तो इसे default

माना जाता है जो public Access Modifier के सामान होता है

2. Protected :

class के protected मेम्बर class के public मेम्बर के सामान होते है protected मेम्बर भी डायरेक्टली start किये जा सकते है और इनकी starting value को सभी subclass के द्वारा इनहेरिट किया जा सकता है यदि टॉप लेवल पर एक मेबर protected है तो इसे इसके class के ऑब्जेक्ट द्वारा एक्सेस किया जा सकता है और साथ ही sub class के ऑब्जेक्ट द्वारा एक्सेस किया जा सकता है

3. friendly :

java में यह default मोडिफायार के रूप में माना जाता है यदि variable method class किसी के भी नाम के साथ किसी भी मोडिफायार का use नहीं किया जाता है तो वह फ्रेंडली type का डिक्लेअर हो जाता है

4. private :

यह access स्पेसीफायर उस class के सभी मेम्बर को private बना देता है जिन पर यह लागु होता है दुसरे शब्दों मेंयदि किसी मेम्बरका एक्सेस मोड private होता है तो तो वह sub class द्वारा use नहीं किया जा सकता है

Abstract Method in java in Hindi

Abstract Method final मेथड के अपोजिट होती है final मेथड यह स्पष्ट करती है की मेथड कभी भी subclass में रिडिफाइन नहीं होती है किन्तु Abstract method कहती है की method subclass में रिडिफाइन होती है अर्थात इसमें ओवरराइडिंग निश्चित होती है Abstract method वह method होती है जो की एक्सीक्यूट होने वाले कोड के आलावा प्रत्येक फॉर्म को डिफाइन करती है एक Abstract method method name,return type ,peramiter type , तथा एक्सेप्शन को डिक्लेअर करती है परन्तु मेथड के लिए execution प्रदान नहीं करती है Abstract method Abstract कीवर्ड के साथ और मेथड की body के स्थान पर सेमीकॉलन के साथ डिक्लेयर की जाती है
इसे निम्लिखित उदाहरण द्वारा दर्शया गया है

```
public abstract void m1();
```

Example :

```
abstract class V
{
    abstract void speed();
}
class Bike extends V
{
    void speed()
    {
        System.out.println("speed limit 30 km / hr");
    }
    public static void main(String[] args)
    {
        V obj=new Bike();
        obj.speed();
    }
}
```

Abstract Class in java

वह class जिसमे एक या एक से अधिक abstract method डिफाइन होती है abstract class कहलाती है abstract class को abstract कीवर्ड के साथ डिक्लेअर किया जाता है एक class को abstract बनाने के लिए abstract डिक्लेयर करना होता है इसमें किसी भी abstract method की उपस्थिति की जरुरत नहीं होती है

abstract class का उदहारण :

```
abstract class Name
{
abstract method 1;
abstract method 2;
}
```

abstract class बनाते समय निम्लिखित बातो का ध्यान रखना चाहये

- किसी भी abstract का ऑब्जेक्ट नहीं बनता है

- एक abstract class की abstract method को उसकी sub class में डिफाइन किया जाता है
- abstract class को static डिफाइन नहीं किया जा सकता है

Example :

```
abstract class V
{
    abstract void speed();
}
class Bike extends V
{
    void speed()
    {
        System.out.println("speed limit 30 km / hr");
    }
    public static void main(String[] args)
    {
        V obj=new Bike();
        obj.speed();
    }
}
```

Array In java

ऐरे एक सामान datatype की value का समूह है अर्थात Array एक ही type के data आइटम का समूह है जो की एक कमान नाम को शेयर करते है ऐरे में उपस्थित variable ऐरे के एलिमेंट कहलाते है ऐरे के एलिमेंट continue memory location पर स्टोर होते है ऐरे के प्रत्येक एलिमेंट index नम्बर के द्वारा एक्सेस किये जाती है ऐरे की index 0 से start होती है

जावा में ऐरे अन्य प्रोग्रामिंग language के भिन्न होता है जावा के ऐरे में c या c++ के तरह ऐरे को डिक्लेअर करते समय ऐरे में एलिमेंट नहीं होते है जावा के ऐरे में डायनामिक मेमोरी allocate की जाती है

java में ऐरे निम्लिखित प्रकार के होते है

1. one Dimension Array
2. Two Dimension Array

One Damnation Array

One Damnation Array का single रो में data स्टोर करने के लिए किया जाता है

Declaration Of One Damnation Array

data_type array name[ ];

Example :

int a[] ;

or

data_type [ ] array_name;

Example :

int [ ]a;

इन दो तरीको से जावा में ऐरे को डिक्लेअर किया जाता है यह पर int ऐरे ला data type है इसका मतलब यह है की ऐरे में केवल integer type के value ही स्टोर की जाएगी और a ऐरे का नाम है

Creation Of Array :

Syntax :

arrayname = new data_type [size];

Example :

a=new int[5];

जावा में ऐरे को क्रिएट करने का मतलब यह होता है की जावा में ऐरे को मेमोरी allocate करना जेसा के हम जानते है जावा में c या c++ के तरह Declaration के समय memory allocate नहीं होती है जावा में ऐरे को memory allocate करने के लिए जावा में ऐरे को क्रिएट करना पड़ता है इस Example में size एट्रिब्यूट ही ऐरे की size हो डिफाइन करता है और इसे ही index नंबर कहते है

Initialization Of Array :

जावा में ऐरे को दो तरह से Initialization किया जाता है

Syntax :

arrayname [index]= value;

Example :

a[0]=”101”;
a[1]=”102”;

Syntax :

data_type array_name[ ] = {list of value};

Example :

int a[ ]={1,2,3,4,5};

Example :

```
import java.util.*;
class Array1
{
	public static void main(String args[])
	{
		Scanner sc=new Scanner(System.in);
		int a[];
		int i;
		a= new int [5];
		System.out.println("Enter 5 Element OF Array :");
		for(i=0;i<5;i++)
		{
			a[i]=sc.nextInt();
		}
		System.out.println("Display Element -");
		for(i=0;i<5;i++)
		{
			System.out.println(a[i]);
		}
	}
}
```

Two Damnation Array

Two Damnation Array का use जावा में रो तथा कॉलम के रूप में data को स्टोर करने के लिए करते है

Declaration Of Two Damnation Array

data_type array_name[ ] [ ] ;

Example :

int a[ ][ ];

Creation Of Array :

Syntax :

array_name =new data_type [size][size];

Example :

a=new int[i][j] ;

यह i और j ऐरे के एलिमेंट की size है i ऐरे की रो को प्रदर्शित करती है और j ऐरे कॉलम को प्रदर्शित करता है

Example of two Damnation Array

```
import java.util.*;
class TwoArray
{
    public static void main(String args[])
    {
        int a[ ][ ];
        a=new int[2][2];
        int i,j;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Element Of Array :");
        for(i=0;i<2;i++)
        {
            for(j=0;j<2;j++)
            {
                a[i][j]=sc.nextInt();
            }
         }
        for(i=0;i<2;i++)
        {
             for(j=0;j<2;j++)
             {
```

```
                System.out.print(a[i][j]);
            }
            System.out.println("");
        }

    }
}
```

String In Java

Java में string कई करैक्टर का समूह होती है जिसे null '\0' नामक स्पेशल करैक्टर के द्वारा समाप्त किया जाता है जावा में किसी भी string को " " double इनवर्टेड कोमा के अन्दर लिखते है java string के लिए ऑब्जेक्ट की तरह व्यवहार करती है जावा में string के लिए एक class भी उपलब्द कराती है जो java.lang package का एक भाग है java string का ऑब्जेक्ट constent String को प्रदर्शित करता है String class में कई मेथड होती है जिनकी सहयता से java programing आसन होती है

जावा में एक बार string ऑब्जेक्ट बना सेने के बाद string करैक्टर बदले नहीं जाते है यधपि string को बनाने के कई तरीके होते है किन्तु string object बनाने के लिए new ऑपरेटर और string class के कंस्ट्रक्टर का use किया जाता है परिवर्तनीय string के लिए StringBuffer class का use किया जाता है जिसके ऑब्जेक्ट में string उपस्थित होती है जो की बनाने के बाद बदली भी जा सकती है String और StringBuffer class java.lang package में उपस्थित होती है और java.lang package सभी प्रोग्राम के लिए स्वत: ही उपस्थित रहती है दोनों class package में final define रहती है अत: इनकी कोई subclass नहीं होती है

इसका उदहारण निम्लिखित है

```
String s=new String ();
```

Method of String Class :

String class java की एक प्रीडिफाइन class है जो java.lang package में डिफाइन है इस

class में निम्लिखित method होती है

1. charAt()

यह मेथड string के अंन्तर्गत किसी स्पेसिफाई करैक्टर को return करती है

2. compareTo()

इस मेथड का use दो string के बिच ने कम्पेरिसन करने के लिए किया जाता है इस मेथड का कार्य दो string के बिच कम्पेरिसन कर के यह चेक करना है की अल्फ़ाबेटिक आर्डर में कोण सी string पहले आएगी

3. concat()

यह मेथड दी गई string को string के अंत में जोड़ती है तथा दोनों string के बिच में स्पेस नहीं होती है

4. equals() :

यह मेथड दो string के बिच में यह परिक्षण करती है की दी गई string equal है या नहीं यदि string बराबर होती है तो यह true return करती है

5. equalignoreCase() :

यह equal मेथड के सामान होती है अंतर केवल इतना है की इसमें case का ध्यान रखने की आवश्यकता नहीं होती है

6. indexof()

यह मेथड स्पेसिफाई करे गए करैक्टर के प्रथम आवर्ती की index return करती है

7.lastindexof()

यह मेथड स्पेसिफाई करे गए करैक्टर के अंतिम आवर्ती की index return करती है

8. length ():

इस मेथड का use string के length पता करने के लिए किया जाता है

9 . replace():

इस मेथड string में स्पेसिफाईड करैक्टर की सभी आवर्तीयो को replace कर देती है

10. toLowerCase()

इस मेथड के द्वारा किसी भी string के सभी करैक्टर को लोअर केस में कन्वर्ट किया जाता है

11. toUpperCase:

यह मेथड किसी भी string के सही characters को अपरकेस में कन्वर्ट कर देती है

String Buffer Class In Java

string Buffer Class म्युटेबल class तथा string class इम्युटेबल class है string Buffer class को डायनामिकली चेज किया जा सकता है इसका use तब किया जाता है जब अधिक मात्र में मॉडिफिकेशन इंसर्शन डीलीशन आदि की आवश्यकता होती है string Buffer class string class के साथ पाई जाती है string data type के साथ फिक्स्ड length की string बनाई जाती है जिसे चेज नहीं किया जा सकता है जबकि string Buffer फेक्सिबल लम्बाई की string बनती है जिसे चेज किया जा सकता है
java में string Buffer class की निम्लिखित method होती है

1. setCharAt()

यह मेथड प्रदान किये गए करैक्टर को स्पेसिफ़ाइड location पर पहले से उपस्थित अन्य करैक्टर को प्रतिस्थापित करती है

2. append() :

यह मेथड सोर्स string के अंत में string को जोडती है

3. insert :

यह मेथड किसी स्पेसल पोजीशन पर string को insert करती है

4. setLength():

यह मेथड सोर्स string की लम्बाई में संशोधन करती है

Vector in java

विभिन language में ऐरे के साथ समस्या यह होती है की ऐरे की size को डायनामिकली बढाया नहीं जाता है अर्थत ऐरे का मान एक बार निर्धरित करने के बाद बदला नहीं जा सकता है परन्तु जावा में यह सम्भव है दुसरे शब्दों में कहे तो जावा में ऐरे का size के बार निर्धरित करने बाद भी बदला जा सकता है इस सामन्य प्रोब्लम को सोल्व करने के लिए एक अन्य जावा आब्जेक्ट कस use किया जाता है यह एक ऐरे के सामान कार्य करता है जिसे वेक्टर ऑब्जेक्ट कहते है अर्थत वेक्टर जावा में एक ऑब्जेक्ट है जो ऐरे की तरह कार्य करता है और अतिरिक कोड लिखने के स्थान पर स्वत : ही इनक्रीस या डिक्रिस हो जाता है . वेक्टर class जावा की library class है जो जावा के java.util package में डिफाइन है एक नए वेक्टर को निम्लिखित प्रकार से डिफाइन किया जाता है

```
Vector Orderlist = new Vector(4);
```

उपरोक्त उदहारण में वेक्टर लिस्ट 4 आइटम के साथ शुरू हुई है ये चार आइटम किसी भी

data type के हो सकते है वेक्टर का आकार रन टाइम पर बढाया जा सकता है

Type of vector list discretion :

Vector को वेक्टर class के कंस्ट्रक्टर के साथ डिक्लेअर किया जाता है वेक्टर class के विभिन argument के विभिन्न कन्स्ट्रक्टर डिफाइन किये गए है जब ये कन्स्ट्रक्टर को allocate करने के लिए use किया जाते है तो तब ये विभिन्न प्रकार के कार्य करने के लिए डिफाइन किये जाते है वेक्टर class के कन्स्ट्रक्टर निम्लिखित है

- Vector()

- Vector(int size)
- Vector(int size,int incr)
- Vector(collection c)

प्रथम कन्स्ट्रक्टर एक डिफ़ॉल्ट वेक्टर बनता है जिसका प्रारम्भिक मान 10 होता है दूसरा कन्स्ट्रक्टर एक वेक्टर बनता है जिसका प्रारम्भिक क्षमता इसके आकर के द्वारा स्पेसिफाई की गई है तीसरा कन्स्ट्रक्टर एक एसा वेक्टर बनता है जिसका जिसकी प्रारम्भिक क्षमता आकार एव इन्क्रीमेंट के द्वारा स्पेसिफाई की गई है चतुर्थ कन्स्ट्रक्टर एक वेक्टर बनता है जिसमे स्टोरेज c के एलिमेंट उपस्थित है

Interface in java

इंटरफ़ेस एक क्लास है जिसमे अन्य क्लास के समान ही डेटा मेम्बर्स तथा मेथड्स उपस्थित होता है इंटरफ़ेस क्लास केवल मेथड्स तथा वेरिएबल को ही डिक्लेअर करती है इंटरफ़ेस class उन सभी मेथड्स जो इन्हें इम्प्लीमेंट करती है को कोमन इंटरफ़ेस प्रदान करती है यह क्लास इन मेथड्स को रन करने के लिए मेथड्स की डिफिनेशन को क्लास के अन्दर लिखती है इंटरफ़ेस के लिए interface keyword का उपयोग किया जाता है इंटरफ़ेस को निम्नलिखित प्रकार से डिक्लेयर किया जाता है –

```
interface Interface_Name
  {
     variable declaration;
     method declaration;
  }
```

इंटरफ़ेस के अन्दर डिक्लेयर होने वाले वेरियेबल को निम्नलिखित प्रकार से डिक्लेयर किया जाता है

Static final variable name= value;

उदाहरण की लिए

Static final int a = 10;

return type method name(para_list);

उदाहरण की लिए

Void show();

Thread in java

थ्रेड एक्सीक्यूट होने वाले प्रोग्राम का सीक्वेंस होता है जो की अन्य थ्रेड के बिना स्वतंत्र रूप से रन होता है तथा दुसरे थ्रेड से डाटा डायरेक्ट शेयर करते है थ्रेड प्रोग्राम के लॉजिकल execution का के प्रथक प्रथक पाथ में ऑर्गेनाइश करने के अनुमति देता है एक थ्रेड स्वय की इंस्ट्रक्शन लिस्ट पर तब तक कार्य करता है जब तक की लिस्ट ख़त्म ना हो जाए या use रुकने को ना कहा जाए समान्य रूप से थ्रेड अलग से रन होने वाला प्रोग्राम होता है

Example :

```
class MyThread extends Thread
{
    public void run()
    {
        System.out.println("Running");
    }
    public static void main(String d[])
```

```
    {
        MyThread my=new MyThread();
        my.start();
    }
}
```

.

Multi Threading in java

किसी भी ऑपरेशन एक समय में एक साथ कई प्रोग्राम को रन किया जा सकता है तथा उन पर कई function भी परफोर्म किये जा सकते है ऑपरेटिंग सिस्टम की इस विशेषता को मल्टीटास्किंग कहते है तथा प्रोग्रामिंग लैंग्वेज में इसे multithreading कहते है जावा language मल्टीथ्रेडेड कांसेप्ट को सोपोर्ट करती है यह language प्रोग्राम को develop करने के लिए control के मल्टिपल फ्लो को बनाने की सुविधा प्रदान करती है प्रत्येक control एक शोर्ट प्रोग्राम होता है जिसे module या थ्रेड कहते है यह थ्रेड समान्तर क्रम में एक्सीक्यूट होते है एक एसा प्रोग्राम जिसमे एक से अधिक कंट्रोल फ्लो होते है मल्टीथ्रेडिंग कहते है किसी भी प्रोग्राम में विभिन्न थ्रेड उपस्थित हो सकते है जो एक साथ कार्य करते है इन गुणो को कोन्क्रेंसी कहते है इस तरह जावा प्रोग्राम में कई छोटे छोटे प्रोग्राम हो सकते है जो की मेन प्रोग्राम के अंतर्गत एक्सीक्यूट होते है और सिस्टम मेमोरी और अन्य रिसोर्स को सयुक्त रूप से उपयोग करते है इन्हें लाइट वेट थ्रेड या लाइट वेट प्रोसेस भी कहते है थ्रेडिंग प्रोग्राम में प्रोसेस के फंक्शन को चित्र में दर्शाया गया है

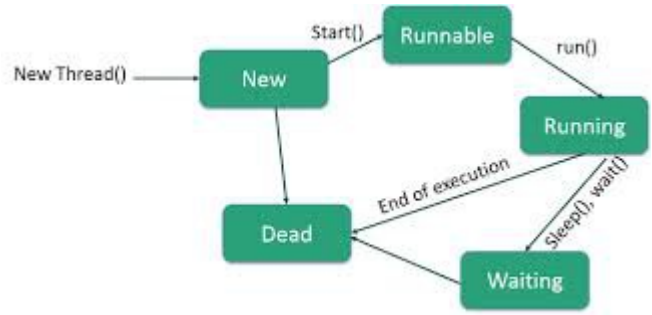
मल्टीथ्रेडिंग की आवश्यकता :

प्रोग्राम को एक समय में केवल एक बार ही रन किया जा सकता है प्रोग्रम के समाप्त होने के बाद दूसरा प्रोग्राम रन किया जाता है इस प्रोसेस को सिंगल थ्रेडेड कहते है लेकिंग एक ही समय में एक से अधिक प्रोग्राम को एक साथ रन करना

मल्टीथ्रेडिंग कहता है मल्टीथ्रेडिंग का use करने के कम समय में अधिक कार्य किया जा सकता है मल्टीथ्रेड का use करने से समय की बचत होती है

life cycle of Thread in java

थ्रेड की लाइफ साइकिल की मल्टिपल स्टेट होते है जिनसे गुजरते हुए एक थ्रेड एक्सीक्यूट होती है एक थ्रेड की life साइकिल चित्र में प्रदर्शित है

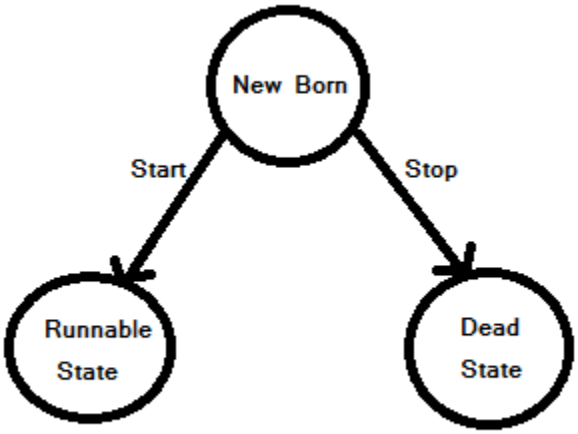


Thread की विभिन्न स्टेट निम्लिखित है

1. न्यू स्टेट :

जब भी किसी थ्रेड को new ऑपरेटर को बनाया जाता है तो इसका मतलब यह होता है की की यह प्रोग्राम new state में है जब तक कोई भी थ्रेड new state में होती है तब तक प्रोग्राम इसके कोड instence को एक्सीक्यूट करने start नहीं करता है

एक new बोर्न थ्रेड का चित्र निम्लिखित है



2. Runnable state :

Runnable का अर्थ यह है की थ्रेड एक्सीक्यूट होने को तेयार है तथा process के free होने का वेट कर रहा है इस तरह थ्रेड थ्रेडेड के उस समूह को join के लेती है जो एक्सीक्यूट होने के लिए processer के free होने का वेट कर रही है इन सभी थ्रेड के एक्सीक्यूट होने के लिए जिस थ्रेड की प्रायोरिटी जादा होती है वह पहले एक्सीक्यूट होती है यदि सभी थ्रेड की प्रायोरिटी सामान है तो थ्रेड के एक्सीक्यूट करने के लिए राउंडरोबिंग का उपयोग किया जाता है अर्थ फर्स्ट काम फर्स्ट सर्व होता है इसका मतलब यह है की जो थ्रेड पहले आया है वह पहले एक्सीक्यूट होता है

3. Running state :

थ्रेड के Running state में होने का मतलब यह होता है की थ्रेड के अन्दर का कोड एक्सीक्यूट हो रही है जब थ्रेड रनिंग स्टेट में होती है तो वह cpu के समय चक्र का use करती है एक थ्रेड Running state में तब होती है जब वह run() मेथड को इनवोक करती है .

Running thread में निम्लिखित 3 मेथड होती है

(a) Suspend() :

इस मेथड का use कर के किसी भी थ्रेड को suspend किया जा सकता है एक suspend थ्रेड को पुन : Running state में लेन के लिए resume() मेथड का use किया जाता है

(b) sleep() :

इस मेथड का use कर के Running thread को निर्धारित टाइम के लिए sleep state में रखते है जब समय पूर्ण हो जाता है तो थ्रेड sleep state से Running state में आ जाती है

(c) Wait():

इस मेथड का use कर के थ्रेड Running state के waiting state में चला जाता है तथा waiting state से यह थ्रेड notify() मेथड का use कर के Running state में लाया जाता है

4. Blocked State :

जब कोई थ्रेड Running state से बाहर होती है तथा किसी कारण से थ्रेड को आगे रन नहीं किया जा सजता है तो इस तरह थ्रेड ब्लॉक्ड स्टेट में आ जाती है ब्लाक थ्रेड वह थ्रेड होती है जो अस्थाई रूप से Running state से बाहर होती है

5. Dead State :

प्रत्येक थ्रेड की एक life cycle होती है जब कोई थ्रेड अपने लिए साइकिल पूरी कर लेती है
तो वह थ्रेड dead state में चली जाती है

stop () Method :

किसी भी थ्रेड को stop करने के लिए stop मेथड का use किया जाता है stop मेथड को निम्लिखित प्रकार से डिक्लेअर किया जाता है
Mythread.stop();
उपरोक्त उदहारण में Mythread एक थ्रेड class है किसी stop किया गया है इसके बाद यह थ्रेड stop state में चली जाती है

Exception Handling in java

Exception को रन टाइम एरर भी कहते है जब किसी प्रोग्राम को रन किया जाता है और उस प्रोग्राम में किसी कारण से रन टाइम कोई एरर है तो use एक्सेप्शन जाता है जावा एक्सेप्शन हैंडलिंग को सपोर्ट कटी है
जावा में एक्सेप्शन को हैंडल करने के लिए try catch का use किया जाता है चटच के ब्लाक में उस कोड को लिखा जाता है जो एक्सेप्शन जनरेट करता हो एक्सेप्शन आने पर try ब्लाक एक्सेप्शन को थ्रो करता है तथा इस थ्रो एक्सेप्शन को catch ब्लाक द्वारा catch किया जाता है
Exception Handling का Example निम्लिखित है

```
import java.util.*;
class Exacption1
{
    public static void main(String args[])
    {
        int a,b,c;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Value Of A :");
        a=sc.nextInt();
```

```
        System.out.println("Enter Value Of B");
        b=sc.nextInt();
        try{
            c=a+b;
            System.out.println("Sum :"+c);
        }
        catch(Exception e)
        {
            System.out.println(e);
        }

    }
}
```

Applet in java

Applet जावा के छोटे – छोटे प्रोग्राम होते है जिनका use कर के हम जावा प्रोग्राम को इंटरनेट पर run करा सकते है applet का use कर के एक जावा प्रोग्राम को मल्टिपल कंप्यूटर पर एक साथ रन किया जाता है जावा में applet को रन करने के लिए applet viewer का use किया जाता है applet प्रोग्राम को वेब ब्राउज़र पर भी रन किया जा सकता है किन्तु applet प्रोग्राम हर किसी ब्राउज़र पर रन नहीं होती है applet प्रोग्राम केवल उन्ही वेब ब्राउज़र पर रन होते है जो applet को सपोर्ट करते है

applet को कंसोल क्लास का use कर के नहीं बनया जाता है applet प्रोग्राम को बनाने के लिए एबस्ट्रेक्ट विंडो टूलकिट (AWT) का use किया जाता है applet का use कर के जावा में GUI प्रोग्रामिंग भी की जा सकती है

java में applet निम्लिखित प्रकार के होती है

1. Remote Applet
2. Local Applet

1. Remote Applet

रिमोट Applet वह applet होते है जिनको स्थनीय कंप्यूटर पर नहीं बनया जाता है रिमोट एप्लेट का use करने के लिए इंटरनेट का use किया जाता है रिमोट एप्लेट का use करने के लिए वेब ब्राउज़र में applet का एड्रेस लिखते है उस address को URL (युनिवर्सल रिसोर्स लोकेटर) कहते है

2. Local Applet

लोकल Applet वह Applet होते है लोकल कंप्यूटर पर डेवलप किया जाता है लोकल एप्लेट को use करने के लिए इन्टरनेट की आवश्यकता नहीं होती है लोकल applet को रन करने के लिए वेब ब्राउज़र के URL (युनिवर्सल रिसोर्स लोकेटर) में उस applet का डायरेक्टरी पाथ दिया जाता है जिससे लोकल Applet रन होता है लोकल Applet कहलाता है

life cycle of applet in java

एक applet प्रोग्राम अपनी पूरी life में बहुत के state से होकर गुजरता है जिसे applet की life cycle कहते है applet प्रोग्राम की life में जो अलग अलग state से होकर गुजरती है तो उन में अलग – अलग मेथड है जिनके कारण हर state का अपना एक यूनिक कार्य होता है applet की life cycle चित्र में बताई गई है

Applet की life cycle में निम्लिखित state होते है

initialization State :

सबसे पहले applet को लोड किया जाता है applet लोड होने के बाद initialization State में आता है जब एक एप्लेट initialization State में होता है applet class में उपस्थित init() मेथड एक्सीक्यूट होती है एक एप्लेट की life में init() मेथड एक बार ही रन होती है

```
public void init()
{
        ……….
        …….(Action)
}
```

Running State :

जब system applet class की start मेथड को call करता है तो applet Running State में चला जाता है start() मेथड तब रन होती है जब applet initialization होता है

start method का syntax निम्लिखित है

```
public void start()
{
   ..........................
   ..........................(Action)
}
```

Display state :

जब applet में किसी प्रोग्राम का आउटपुट डिस्प्ले करना है तो applet display state में चला जाता है Display state में जाने के बाद applet class के paint() मेथड एक्सीक्यूट होती है

paint () Method का syntax निम्लिखित है

```
public void paint()
{
   ..................
   ..................(Action)
}
```

idle state :

जब किसी applet को use करते - करते अचनक बंद कर दिया जाता है तब applet idle state में चला जाता है applet को idle state के ले जाने के लिए applet class की stop मेथड का use भी किया जाता है
stop() का syntax निम्लिखित है

```
public void stop()
{
   .................
   .................(Action)
}
```

Dead State :

जब किसी applet को memory से remove होती है तो applet dead state में चला जाता है जब applet dead state में जाता है तो applet class की destroy () मेथड autometic एक्सीक्यूट होती है

destroy method का syntax निम्लिखित है

```
public void destroy()
{
   ............
   ............(Action)
}
```

How To Make Applet Program :

जेसा को हम जानते है की applet जावा का एक एसा प्रोग्राम होता है जो वेब ब्राउजर पर रन होता है इस लिए हमें जावा के प्रोग्राम को रन करने के लिए वेब

ब्राउजर की आवश्यकता होती है

एक applet प्रोग्राम को बनाने के लिए आपको निम्लिखित step का पालन करने की आवश्यकता होती है

सबसे पहले आपको एक .जावा फाइल बनानी होती है जिसमे आप applet का कोड लिखते है

उदहारण के लिए आपके द्वारा बनाई गई जावा फाइल Example1.java है

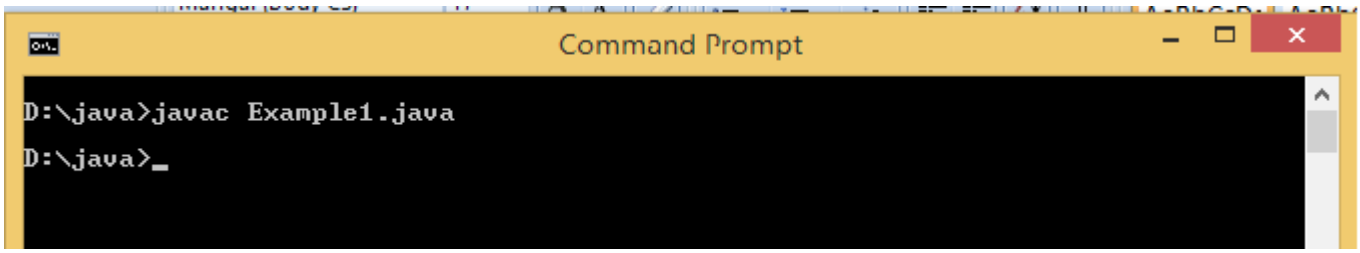
हम Example1.java फाइल में applet का कोड लिखेगे तथा उसे compile कर के उसकी class फाइल क्रिएट करेगे

File Name : Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example extends Applet
{
   public void paint(Graphics g)
   {
      g.drawString("Hello",50,50);
   }
}
```

उपरोक्त Example में हमने एक string को screen पर display करने के लिए drawString() का use कर के एक प्रोग्राम बनया है

अब इस फाइल को compile करेगे



इस फाइल को compile करने के बाद हमें एक class फाइल प्राप्त होगी उस class फाइल का नाम Example1.class है

इसके बाद हम एक new HTML फाइल क्रिएट करेगे उस HTML फाइल का नाम Ex.html है

अब हम इस ex1.html में हम Example1.class फाइल को add करेगे
इस new html फाइल में हम <applet></applet> tag का उपयोग करेगे
इस applet tag के निम्नलिखित एट्रिब्यूट होते है

1. code :

इस एट्रिब्यूट में हम applet प्रेग्राम की class फाइल का नम देगे जेस की हमरे प्रेग्राम में class फाइल Example1.class है ते code एट्रिब्यूट में इसका नम देगे

```
<applet code="Example1"></applet>
```

2. width :

width एट्रिब्यूट का use कर के applet width set करते है

3 .height :

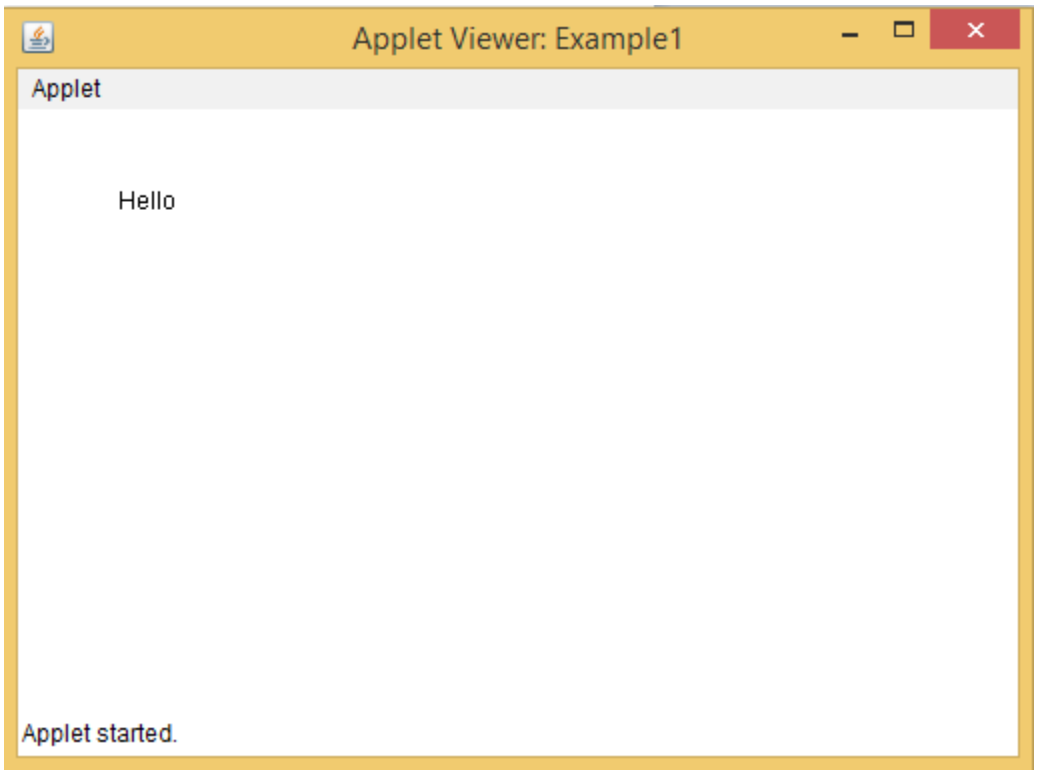
height एट्रिब्यूट का use कर के हम applet की height set करेगे

applet प्रोग्राम को रन करने के लिए HTML फाइल क्रिएट करने का कोड

```
<!DOCTYPE html>
<html>
<head>
```

```
   <title>Applet Test</title>
</head>
<body>
  <applet code="Example1" width="500" height="300"></applet>
</body>
</html>
```

अब इस फाइल को रन करने के लिए applet viewer का use करेगे



drawString() Method In Java :

drawString एक मेथड है यह Graphics क्लास को एक मेथड है जिसका उपयोग कर के applet पर string को print करने के लिए करते है यह मेथड तिन peramiter लेती है इसकें पहला परमिटर String होती है तथा दो parameter x तथा y की value होती है

Syntax :

drawstring(“string”,x,y);

write a program using drawstring method

File Name : Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example extends Applet
{
    public void paint(Graphics g)
    {
        g.drawString("Hello",50,50);
    }
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Applet Test</title>
</head>
<body>
   <applet code="Example1" width="500" height="300"></applet>
```

```
</body>
</html>
```

drawLine () :

इस मेथड का use applet पर line draw करने के लिए किया जाता है drawLine मेथड graphics class की मेथड है यह मेथड 4 पेरामीटर लेती है

syntax :

```
drawLine(x1,y1,x2,y2);
```

Write A Program Using drawLine Method

Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
	public void paint(Graphics g)
	{
	  g.drawLine(10,10,50,50);
	}
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
	<title>Applet Test</title>
</head>
<body>
	<applet code="Example1" width="300" height="300"></applet>
</body>
</html>
```

drawRect() Method In Java :

इस मेथड का use applet पर rectangle draw करने के लिए करते है यह मेथड Graphics class की मेथड है यह मेथड 4 पेरामीटर लेती है

Syntax :

drawRect(x,y,width,height);

write a Program using drawRect method

File Name : Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
        public void paint(Graphics g)
        {
            g.drawRect(50,100,200,150);
        }
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Applet Test</title>
</head>
<body>
   <applet code="Example1" width="500" height="300"></applet>
</body>
</html>
```

fillRect() Method :

यह मेथड drawRect की तरह ही होती है इसका use कर कर rectangle बनाने के लिए किया जाता है किन्तु इस मेथड का उस कर कर Fill rectangle बनया जाता है और इस Rectangle में by default Color black होता है

syntax :

fillRect(x,y,width,height);

Write a Program Using fillRect() method

Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
	public void paint(Graphics g)
	{
	  g.fillRect(50,100,200,150);
	}
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
	 <title>Applet Test</title>
</head>
<body>
	<applet code="Example1" width="300" height="300"></applet>
```

```
</body>
</html>
```

drawRoundRect() Method In Java

इस मेथड का use कर के rectangle बनाया जाता है इस मेथड के द्वारा जो rectangle बनाया जाता है वह border से rounded होता है

Syntax :

drawRoundRect(x,y,width,height,round-left,round-right);

Write a program Using drawRoundRect method

File Name : Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
        public void paint(Graphics g)
        {
            g.drawRoundRect(10,100,80,50,10,10);
        }
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Applet Test</title>
</head>
<body>
   <applet code="Example1" width="500" height="300"></applet>
```

```
</body>
</html>
```

fillRoundRect() Method :

Write a Program Using fillRect() method

Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
        public void paint(Graphics g)
        {
            g.fillRoundRect(10,100,80,50,10,10);
        }
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
         <title>Applet Test</title>
</head>
<body>
        <applet code="Example1" width="300" height="300"></applet>
</body>
</html>
```

drawOval() Method In Java :

इस मेथड का उस कर के applet पर oval draw किया जाता है यह मेथड Graphics class की है यह मेथड 4 पेरामीटर लेती है

Syntax :

drawOval(x,y,width,height);

write a program Using drawOval() method

File Name : Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
        public void paint(Graphics g)
        {
            g.drawOval(50,100,150,100);
        }
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Applet Test</title>
</head>
<body>
   <applet code="Example1" width="500" height="300"></applet>
</body>
</html>
```

fillOval() Method :

यह मेथड drawOval के सामान ही होती है किन्तु इस मेथड का use के जो Oval बनाया जाता है वह filled होता है by default oval में black color fill होता है

syntax :

fillOval(x,y,width,height);

write a program using fillOvel() method

Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
	public void paint(Graphics g)
	{
	   g.fillOval(50,100,150,100);
	}
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
	 <title>Applet Test</title>
</head>
<body>
	<applet code="Example1" width="300" height="300"></applet>
</body>
</html>
```

setColor()Method In Java :

इस मेथड का use कर के applet के किसी भी ऑब्जेक्ट में color फिल किया जाता है

Syntax :

setColor(Color.colorname);

write a Program Using setColor() method

File Name : Example1.java

```
import java.applet.*;
import java.awt.*;
public class Example1 extends Applet
{
        public void paint(Graphics g)
        {
            g.setColor(Color.red);
            g.fillOval(50,100,150,100);
        }
}
```

ex1.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Applet Test</title>
</head>
<body>
   <applet code="Example1" width="500" height="300"></applet>
</body>
</html>
```

What Is JDBC

JDBC का full From Java Database Connectivity है इसको javasoft द्वारा डेवलप किया गया था JDBC एक API है जिसका उस कर के जावा को डेटाबेस से कनेक्ट किया जाता है JDBC का use कर के डेटाबेस को टेबल के रूप में प्रदर्शित किया जाता है किसी भी computer पर जावा को डेटाबेस से कनेक्ट करने के लिए उस कंप्यूटर में JDBC का ड्राइवर होना आवश्यक है जिसका use के जावा को डेटाबेस से कनेक्ट किया जाता है जावा को डेटाबेस के कनेक्ट करने के लिए कुछ मेथड तथा इंटरफ़ेस का use किया जाता है जिनकी सहयता से डेटाबेस से कनेक्शन स्थापित किया जाता है जिन मेथड और इंटरफ़ेस का use कर के कनेक्शन स्थापित किया जाता है ये सभी java.sql package में होती है इसको import कर के डेटाबेस से कनेक्शन स्थापित किया जाता है

JDBC Connection को स्थपित करने के लिए प्रमुख इंटरफ़ेस

- Connection
- PreparedStatement
- Resultset

JDBC Connection को स्थपित करने के लिए प्रमुख class

- DriverManager
- SQLException

JDBC connection के लिए प्रमुख मेथड निम्लिखित है

- Class.forName(“driver name”);

- DriverManager.getConnection("url","username","password") ;

1. Class.forName() :

इस मेथड का use ड्राइवर को memory में लोड करने के लिए किया जाता है Class.forName() मेथड में एक पेरामीटर पास किया जाता है Driver name इस method के पेरामीटर में उस ड्राइवर का नाम pass किया जाता है जिससे हमें कनेक्शन स्थापित करना है
उदहारण के लिए यदि हमें mysql से कनेक्शन स्थापित करना है तो हमें निम्लिखित तरीके के ड्राइवर का नाम लिखना होगा

Example :

2. DriverManager.getConnection() :

इस मेथड में 3 पेरामीटर पास होते है url , user name और password इस मेथड को उदाहरण के रूप में समझते है यदि हमें mysql से कनेक्शन स्थापित करना है तो किस तरह से लिखेगे

Syntax :

DriverManager.getConnection("url","username","passsword");

Example :

DriverManager.get.Connection("jdbc:mysql:\\localhost\dbname","root","123456");

3. prepareStatement ('query') :

यह Connection interface की एक method है इसका use sql कमांड को execute करने के किया जाता है यह method एक parameter लेती है तथा

sql कमांड को एक्सीक्यूट करके उसके रिजल्ट को return करती है जिसे preparedStatement statement interface के reference variable में स्टोर की जाती है

syntax :

pst = prepareStatement(“query”);

Example :

pst = prepareStatement(“select * from student”);

यह पर pst PreparedStatement interface का रिफरेन्स है

How To Connnect java To Mysql Database

```
import java.sql.*;
class Database
{
	public static void main(String d[])
	{
		Connection con;
		PreparedStatement pst;
		ResultSet rs;
		try{

Class.forName("com.mysql.jdbc.Driver").newInstance();

con=DriverManager.getConnection("jdbc:mysql://localhost/college","root","");
                                        if(!con.isClosed())
                                        {
                                                System.out.println("Connect");
                                        }
                                        else
                                        {
                                                System.out.println("Not
Connnect");
                                        }
```

```
                                        }
                                        catch(Exception e)
                                        {
                                                System.out.println(e);
                                        }
                }
}
```

How To Connnect java To Orecal Database

```
import java.sql.*;
class Database
{
        public static void main(String d[])
        {
                Connection con;
                PreparedStatement pst;
                ResultSet rs;
                try{
                                Class.forName("Oracle.Jdbc.OracleDriver
").newInstance();

con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe
","system","123456");
                                if(!con.isClosed())
                                {
                                        System.out.println("Connect");
                                }
                                else
                                {
                                        System.out.println("Not Connnect");
                                }
                        }
                        catch(Exception e)
                        {
                                System.out.println(e);
                        }
        }
}
```

.

What Is File Handling

जब भी हम जावा में कोई प्रोग्राम बनाते है तो उस प्रोग्राम में हम किसी ना किसी प्रकार के data का use करते है वह data तब तक ही कंप्यूटर की memory में रहता है जब तक प्रोग्राम एक्सीक्यूट होता है जेसे ही प्रोग्राम एंड होता है डाटा कंप्यूटर की memory से नष्ट हो जाता है
यदि हमें अगली बार data की आवश्यकता होती है तो हमें वह data नहीं मिल पाता है इस समस्या को खतम करने के लिए जावा में फाइल हेंडलिंग का use किया जाता है जावा में फाइल हेंडलिंग का use के जावा प्रोग्राम में use किया जाने वाला data फाइल के रूप में स्टोर किया जाता है ताकि बाद में प्रोग्राम के end होने के बाद भी यदि हमें उस data की आवश्यकता होती है तो उसे हम use कर सकते है
दुसरे शब्दों में कहे तो जावा में data को परमानेंट फाइल के रूप में स्टोर करने के लिए फाइल हेंडलिंग का उपयोग किया जाता है

File Class In Java

जेसा की हम जानते है की जावा में फाइल हेंडलिंग का use data को स्टोर करने के लिए किया जाता है data को फाइल में स्टोर किया जाता है उस फाइल को reprsent के लिए जावा में File class का use किया जाता है File class java.io package में है
File class की प्रमुख मेथड निम्लिखित है

1. **exists()**

इस मेथड का use यह पता करने कल लिए किया जाता है की File exists है या नहीं

2. **length() :**

इस मेथड का use कर के फाइल की size को पता किया जाता है

3. getName() :

इस मेथड का use कर के फाइल की नाम को पता किया जाता है

4. canWrite() :

इस मेथड का use कर के यह पता किया जाता है की फाइल में data write करने की परमिशन है या नहीं

File class का use करके फाइल पर परफॉर्म किये जाने वाले ऑपरेशन प्रोग्राम मे बताए गए है

```
import java.io.*;
class FileExample
{
    public static void main(String args[]) throws IOException
    {
        File f1=new File("file1.txt");
        f1.createNewFile();
        System.out.println("File Existe :"+f1.exists());
        System.out.println("File Length :"+f1.length());
        System.out.println("File Name :"+f1.getName());
        System.out.println("Can Write :"+f1.canWrite());
    }
}
```

FileOutputStream Class

यह class java.io package में है इसका use कर के data को फाइल में राइट किया जाता है इस class में निम्लिखित कंस्ट्रक्टर होते है जिनक उस कर के फाइल में data को write किया जाता है

FileOutputStream class के प्रमुख कंस्ट्रक्टर निम्लिखित है

1. FileOutputStream fout=new FileOutputStream("file path");

इस प्रकार के कंस्ट्रक्टर में एक parameter पास किया जाता है फाइल का पाथ

फाइल के पाथ से तात्पर्य यह है की जिस फाइल में data write करना है उस फाइल की location बताई जाती है की वह फाइल कहा पर है उसका पाथ दिया जाता है

2. FileOutputStream fout=new FileOutputStream("file path","boolen value");

इस प्रकार के कंस्ट्रक्टर में दो parameter पास किया जाता है

- File का Path
- एक Boolen value (true या false)

फाइल के पाथ से तात्पर्य यह है की जिस फाइल में data write करना है उस फाइल की location बताई जाती है की वह फाइल कहा पर है उसका पाथ दिया जाता है

तथा दुसरे parameter में boolen value से तात्पर्य यह है की उसमे true या false दोनों में से किसी एक को लिखा जाता है यहा

true का मतलब यह होता है की फाइल में data को append करना है और यादी data को append नहीं करना होता है तो false लिखा जाता है यदि हम इन दोनों में से कुछ भी नहीं लिखते है तो by default false माना जाता है

FileoutputStream का use करते हुए फाइल में data को write करना

```
import java.io.*;
class Fos
{
  public static void main(String args[]) throws IOException
  {
       int i;
       FileOutputStream fout=new
FileOutputStream("file1.txt",true);
       String s="D.S Rajput";
       char ch[]=s.toCharArray();
       for(i=0;i<s.length();i++)
       {
            fout.write(ch[i]);
       }
       fout.close();
  }
}
```

FileInputStream Class In Java

यह class java.io package में होती है इसका use फाइल से data को रीड करने के लिए किया जाता है इस class में निम्लिखित parameter होते है जिनका उपयोग कर के फाइल से data को रीड किया जाता है

1. FileInputStream fin=new FileInputStream (“file path”);

इस प्रकार के कंस्ट्रक्टर में एक parameter पास किया जाता है फाइल का पाथ

फाइल के पाथ से तात्पर्य यह है की जिस फाइल से data read करना है उस फाइल की location बताई जाती है की वह फाइल कहा पर है उसका पाथ दिया जाता है

FileInputStream का use करते हुए फाइल से data को read करना

```
import java.io.*;
class Fis
{
    public static void main(String d[]) throws IOException
    {
        FileInputStream fin=new FileInputStream("file1.txt");
        int i=1;
        while(i!=-1)
        {
            i=fin.read();
            if(i!=-1)
            {
                System.out.print((char)i);
            }
        }
        fin.close();
    }
}
```