JAVA

"Door Of Your Dreams"

KR NETWORK CLOUD

Leading Edge IT Training

Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language.

Java was developed by *Sun Microsystems* (which is now the subsidiary of Oracle) in the year 1995. *James Gosling* is known as the father of Java. Before Java, its name was *Oak*. Since Oak was already a registered company, so James Gosling and his team changed the Oak name to Java.

1. Desktop Applications such as acrobat reader, media player, antivirus, etc.
2. Web Applications
3. Enterprise Applications such as banking applications.
4. Mobile
5. Embedded System
6. Smart Card
7. Robotics
8. Games, etc.

Java Version History

Many java versions have been released till now. The current stable release of Java is Java SE 10.

1. JDK Alpha and Beta (1995)
2. JDK 1.0 (23rd Jan 1996)
3. JDK 1.1 (19th Feb 1997)
4. J2SE 1.2 (8th Dec 1998)
5. J2SE 1.3 (8th May 2000)
6. J2SE 1.4 (6th Feb 2002)
7. J2SE 5.0 (30th Sep 2004)
8. Java SE 6 (11th Dec 2006)
9. Java SE 7 (28th July 2011)
10. Java SE 8 (18th Mar 2014)
11. Java SE 9 (21st Sep 2017)
12. Java SE 10 (20th Mar 2018)

Platform Independent

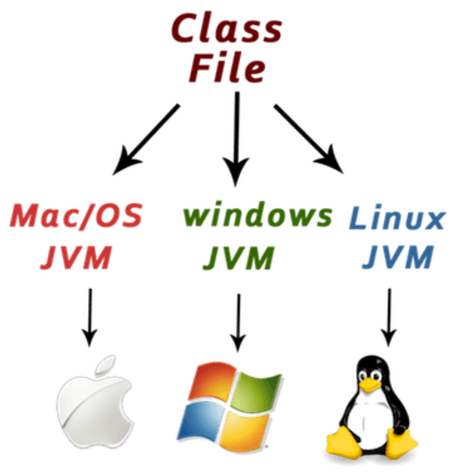
Java is platform independent because it is different from other languages like C, C++, etc. which are compiled into platform specific machines while Java is a write once, run anywhere language. A platform is the hardware or software environment in which a program runs.

There are two types of platforms software-based and hardware-based. Java provides a software-based platform.

The Java platform differs from most other platforms in the sense that it is a software-based platform that runs on the top of other hardware-based platforms. It has two components:

1. Runtime Environment
2. API(Application Programming Interface)

Java code can be run on multiple platforms, for example, Windows, Linux, Sun Solaris, Mac/OS, etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform-independent code because it can be run on multiple platforms, i.e., Write Once and Run Anywhere(WORA).



Object-oriented

Java is an object-oriented programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior.

Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.

Basic concepts of OOPs are:

1. Object
2. Class
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation

First Java Program

```
Class atul

{

        public static void main(String s[])

        {

                System.out.print(“Hello”);

        }

}
```

Install the JDK if you don't have installed it

Set path of the jdk/bin directory.

Create the java program.

Save it as :- atul.java

Compile :- javac atul.java

Run:- java atul (run it with class file name).

Identifiers

Identifiers are user defined names given to variables name ,function name etc. in order to refer in program.

- The first character of an identifiers should start with an alphabet or underscore.
- The identifiers should contain alphabets, digits and underscore.
- There should not be any whitespace within the identifier.
- Keywords should not be used.

Ex:-

Valid	Invalid
a	1a
Atul	1atul
A1	Atul
Atul1	
Atul_kumar	

Keywords

Reserved word of any programming language is called Keywords.

Ex:- int, for,while,char,if etc.

Comments

Comments are not an executable part but a comment just describes the program or provides information about each statements in the program.

Single Line Comment

// Comment

Multiline Comment

/** Comment */

Data Type

What type of data you want to store in the program.

There are three type of data type we can use.

1)Primary data type

2)Derived data type

3)User-defined data type

1)Primary data type:-

The data type that represent numbers and characters. Integer, float and character.

2)Derived data type:-

The data type that are obtained from the primary data type such as array and structures.

3)User-defined data type:-

Data type that are defined by the user like function.

Data Type	**Default Value**	**Default size**
boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte

short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Typecasting

```
class Simple
{
        public static void main(String[] s)
        {
                float f=10.5f;
                //int a=f;//Compile time error
                int a=(int)f;
                System.out.println(f);
                System.out.println(a);
        }
}
```

P1) write a program to print your name.

```
class atul
{
        public static void main(String s[])
        {
                System.out.print("Atul Kumar Varshney");
        }
}
```

P2) write a program to add two numbers..

```
class atul
```

```
{
        public static void main(String s[])
        {
                int a=10,b=20,c;
                c=a+b;
                System.out.print ("Sum"+c);
        }
}
```

P3) Write a program to multiply two values.

```
class atul
{
        public static void main(String s[])
        {
                int a=10,b=2,c;
                c=a*b;
                System.out.print ("Mul="+c);
        }
}
```

P4) Write a program to sub two values.

```
class atul
{
        public static void main(String s[])
        {
                int a=10,b=2,c;
                c=a-b;
                System.out.print("Sub="+c);
        }
}
```

P5) Write a program to divide two values.

```
class atul
{
        public static void main(String s[])
        {
                int a=10,b=2;
                float c;
                c=a/b;
                System.out.print ("Div="+c);
        }
}
```

P6) Write a program to add two value input by user.

```
import java.io.*;
class atul
{
        void add()throws Exception
        {
                int a,b,c;
                System.out.print("Enter two value=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                c=a+b;
                System.out.print("Add:-"+c);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob.add();
        }
}
```

P7) Write a program to sub two value input by user.

```
import java.io.*;
class atul
{
	void sub()throws Exception
	{
		int a,b,c;
		System.out.print("Enter two value=");
		BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
		a=Integer.parseInt(br.readLine());
		b=Integer.parseInt(br.readLine());
		c=a-b;
		System.out.print("Sub:-"+c);
	}
	public static void main(String s[])throws Exception
	{
		atul ob=new atul();
		ob.sub();
	}
}
```

P8) Write a program to mul two values value input by the user.

```
import java.io.*;
class atul
{
	void mul()throws Exception
	{
		int a,b,c;
		System.out.print("Enter two value=");
		BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
		a=Integer.parseInt(br.readLine());
```

```
                b=Integer.parseInt(br.readLine());
                c=a*b;
                System.out.print("Mul:-"+c);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob.mul();
        }
}
```

P9) Write a program to divide two value values input by the user.

```
import java.io.*;
class atul
{
        void div()throws Exception
        {
                int a,b,c;
                System.out.print("Enter two value=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                c=a/b;
                System.out.print("div:-"+c);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob.div();
        }
}
```

P10) Write a program to calculate the area to circle.

```
import java.io.*;
class atul
{
        void area()throws Exception
        {
                int rad;
                float area;
                System.out.print("Enter a rad=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                rad= Integer.parseInt(br.readLine());
                area=3.14*rad*rad;
                System.out.print("Area of Circle="+area);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. area();
        }
}
```

P11) Write a program to calculate the area of Rect.

```
import java.io.*;
class atul
{
        void area()throws Exception
        {
                int height,width,area;
                System.out.print("Enter the Height=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                height= Integer.parseInt(br.readLine());
                System.out.print("Enter the width=");
                width= Integer.parseInt(br.readLine());
                area=height*width;
                System.out.print("Area of Rec="+area);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. area();
        }
}
```

P12) Write a program to calculate the area of square.

```
import java.io.*;
class atul
{
        void area()throws Exception
        {
                int rad,area;
                System.out.print("Enter the rad=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                rad=Integer.parseInt(br.readLine());
                area=rad*rad;
                System.out.print("Area of squ="+area);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. area();
        }
}
```

P13) Write a program to find ASCII value of any character.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                char ch;
                System.out.print("Enter any character=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                ch=(char)br.read();
                System.out.print("The ASCII value of any character="+&ch);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P14) Write a program to Compute Quotient and Remainder.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int val1,val2,Quotient,Remainder;
                System.out.print("Enter two values:-");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                val1=Integer.parseInt(br.readLine());
                val2=Integer.parseInt(br.readLine());
                Quotient=val1/val2;
                System.out.print("Quotient=%d",Quotient);
                Remainder=val1%val2;
                System.out.print("Remainder=%d",Remainder);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P15) Write a program to find the size of a variable.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int a;
                float b;
                char c;
                double d;
                System.out.print("Size of int data type:-"+sizeof(a)+"\n");
                System.out.print("Size of float data type:-"+sizeof(b) +"\n");
                System.out.print("Size of char data type:-"+sizeof(c) +"\n");
                System.out.print("Size of double data type:-"+sizeof(d) +"\n");
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P16) Write a program to swap two values by using third variable.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int a,b,t;
                System.out.print("Enter two values=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                System.out.print("Before swap the value of a=”+a+”and the value of b="+b);
                t=a;
                a=b;
                b=t;
                System.out.print("After swap the value of a=”+a+”and the value of b=%d"+b);

        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P17) Write a program to swap two value without using third variable.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int a,b;
                System.out.print("Enter two values=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                System.out.print("Before swap the value of a=”+a+”and the value of b="+b);
                a=a+b;
                b=a-b;
                a=a-b;
                System.out.print("After swap the value of a=”+a+”and the value of b=%d"+b);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

Operators

An operator is a symbol that operates on a value or a variable. For example: `+` is an operator to perform addition.

C has a wide range of operators to perform various operations.

1) **Arithmetic Operators**

+	Add two or more values
-	Sub two or more values
*	mul two or more values
/	div two or more values
%	Remainder of two values

P18) Write a program to perform a Arithmetic Operators.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int a,b,add,sub,mul,div,rem;
                System.out.print("Enter two values=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                add=a+b;
                sub=a-b;
                mul=a*b;
                div=a/b;
                rem=a%b;
                System.out.print("Add of two value="+add);
                System.out.print("\nSub of two value="+sub);
                System.out.print("\nMul of two value="+mul);
                System.out.print("\nDiv of two value="+div);
                System.out.print("\nRem of two value="+rem);
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

2) <u>Assignment Operators</u>

=	Assign a value
+=	
-=	
*=	
/=	

P19) Write a program to perform Assignment Operators.

```
import java.io.*;
class atul
{
	void ass()throws Exception
	{
		int n,a;
		printf("Enter any value=");
		BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
		n=Integer.parseInt(br.readLine());
		a=n;
		System.out.print("Assign a value to a="+a);
		n+=2;
		System.out.print("\nThe value of n increment by 2 after increment the
value of n="+n);
		n-=3;
		System.out.print("\nThe value of n decrement by 3 after decrement the
value of n="+n);
		n*=2;
		System.out.print("\nThe value of n multiply by 2 after multiply the value of
n="+n);
		n/=2;
		System.out.print("\nThe value of n divide by 2 after dividation the value of
n="+n);
	}
	public static void main(String s[])throws Exception
	{
```

```
                atul ob=new atul();
                ob. ass();
        }
}
```

3)Relational Operators

==	comp two values
>	Grater then
<	Less then
>=	Grater then and equal to
<=	Less then and equal to
! =	not equal to

P20) Write a program to define Relational Operators.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int a,b;
                System.out.print("Enter two values=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                System.out.print("1 means true and 0 means false");
                System.out.print("\nThe value of a and b are equal or not"+(a==b));
                System.out.print("\nThe value of a and b are grater then or not"+(a>b));
                System.out.print("\nThe value of a and b are less then or not"+(a<b));
                System.out.print("\nThe value of a and b are grater then equal to or not
"+(a>=b));
                System.out.print("\nThe value of a and b are less the equal to or not
"+(a<=b));
                System.out.print("\nThe value of a and b are not equal"+(a!=b));
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

4)Logical Operators

&&	And operators
||	or operator

And(&&) operator:-

Condition 1	Condition 2	Result
T	T	T
T	F	F
F	T	F
F	F	F

Or (||) Operators:-

Condition 1	Condition 2	Result
T	T	T
T	F	T
F	T	T
F	F	F

Controlling Statements

Types of Controlling Statements

1) If
2) If else
3) Nested if else
4) Else if
5) Switch

1)<u>if Statement</u>:-

Syntax:-

```
if(condition)
{
        Statements;
}
```

P21) Write a program to check that the given value is grate then 10 print your name.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter any value=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                if(n>10)
                {
                        System.out.print("Atul");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

2)<u>if else Statements</u>:-

Syntax:-

```
                If(condition)
                {
                        Statements;
                }
                else
                {
                        Statements;
                }
```

P22) write a program to check that the give value is even and odd.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter any value=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                if(n%2==0)
                {
                        System.out.print("Value is even");
                }
                else
                {
                        System.out.print("value is odd");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P23) Write a program to check that the give character is vow or cons.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                char ch;
                System.out.print("Enter any character=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                ch=(char)br.read();
                if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u')
                {
                        System.out.print("The given char is vow");
                }
                else
                {
                        System.out.print("The given char is con");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P24) write a program if the sugar rate is grater then 60 print ½ kg other wise 1 kg.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                if(n>60)
                {
                        System.out.print("1/2 KG");
                }
                else
                {
                        System.out.print("1 KG");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P25) Write a program to check that the grater value b/w two variables.

```
import java.io.*;
class atul
```

```
{
        void ass()throws Exception
        {
                int val1,val2;
                System.out.print("Enter two values=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                val1=Integer.parseInt(br.readLine());
                val2=Integer.parseInt(br.readLine());
                if(val1>val2)
                {
                        System.out.print("Value 1 is Grater");
                }
                else
                {
                        System.out.print("Value 2 is Grater");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

3) <u>Nested if else</u>:-

Syntax:-

```
        if(condition)
        {
                If(condition)
                {
                        If(condition)
                        {
                                Statements;
                        }
                        else
                        {
                                Statements;
                        }
                }
                else
                {
                        Statements;
                }
        }
        else
        {
                If(condition)
                {
                        Statements;
                }
                else
                {
                        Statements;
                }
        }
```

P26) Write a program to check that the grate value b/w three variables.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
```

```
        int val1,val2,val3;
        System.out.print("Enter three values:-");
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        val1=Integer.parseInt(br.readLine());
        val2=Integer.parseInt(br.readLine());
        val3=Integer.parseInt(br.readLine());
        if(val1>val2)
        {
                if(val1>val3)
                {
                        System.out.print("Value 1 is grater");
                }
                else
                {
                        System.out.print("Value 3 is grater");
                }
        }
        else
        {
                if(val2>val3)
                {
                        System.out.print("Value 2 is grater");
                }
                else
                {
                        System.out.print("Value 3 is grater");
                }
        }
}
```

```
          public static void main(String s[])throws Exception
          {
                    atul ob=new atul();
                    ob. ass();
          }
}
```

4)**<u>else if Statements</u>**:-

Syntax:-

```
if(condition)
{
          Statements;
}
else if(condition)
{
          Statements;
}
else if(condition)
{
          Statements;
}
else if(condition)
{
          Statements;
}
.
.
.
else
```

```
{
        Statements;
}
```

P27) Write a program to enter a week no and then print the related week day.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter a week no:-");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                if(n==1)
                {
                        System.out.print("Sunday");
                }
                else if(n==2)
                {
                        System.out.print("Monday");
                }
                else if(n==3)
                {
                        System.out.print("Tuesday");
                }
                else if(n==4)
                {
                        System.out.print("Wednesday");
                }
                else if(n==5)
                {
                        System.out.print("Thuresday");
                }
                else if(n==6)
                {
                        System.out.print"Friday");
                }
                else if(n==7)
                {
                        System.out.print("Saturday");
                }
                else
                {
                        System.out.print("!Wrong Week No");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P28) Write a program to enter a month no and then print the month name.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
```

```
System.out.print("Enter a week no:-");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
n=Integer.parseInt(br.readLine());
if(n==1)
{
        System.out.print("Jan");
}
else if(n==2)
{
        System.out.print("Feb");
}
else if(n==3)
{
        System.out.print("Mar");
}
else if(n==4)
{
        System.out.print("Apr");
}
else if(n==5)
{
        System.out.print("May");
}
else if(n==6)
{
        System.out.print("Jun");
}
else if(n==7)
{
```

```
                System.out.print("july");
        }
        else if(n==8)
        {
                System.out.print("Aug");
        }
        else if(n==9)
        {
                System.out.print("Sep");
        }
        else if(n==10)
        {
                System.out.print("Oct");
        }
        else if(n==11)
        {
                System.out.print("Nov");
        }
        else if(n==12)
        {
                System.out.print("Dec");
        }
        else
        {
                System.out.print("!Wrong Month no.");
        }
}
public static void main(String s[])throws Exception
{
```

```
                atul ob=new atul();
                ob. ass();
        }
}
```

P29) Write a program to create a simple calculater by using else if.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n,a,b,c;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("1.Add\n2.Sub\n3.Mul\n4.Div\n");
                System.out.print("Enter your choice=");
                n=Integer.parseInt(br.readLine());
                System.out.print("Enter two value=");
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                if(n==1)
                {
                        c=a+b;
                        System.out.print("Sum="+c);
                }
                else if(n==2)
                {
                        c=a-b;
                        System.out.print("Sub="+c);
                }
                else if(n==3)
                {
                    c=a*b;
                    System.out.print("Mul="+c);
          }
          else if(n==4)
          {
                    c=a/b;
                    System.out.print("Div="+c);
          }
          else
          {
                    System.out.print("!Wrong Choice");
          }
     }
     public static void main(String s[])throws Exception
     {
          atul ob=new atul();
          ob. ass();
     }
}
```

5)**<u>Switch Case</u>**:-

Syntax:-

```
switch(vari)
{
          case 1:   Statements;
                     break;
          case 2:   Statements;
                    break;
          case 3:   Statements;
                    break;
```

```
                case 4:   Statements;
                            break;
                .
                .
                .
                .
                .
                default :     statements;
        }
```

P30) Write a program to enter a week no and print the related week name.

```
import java.io.*;
class atul
{
    void ass()throws Exception
    {
        int n;
        System.out.print("Enter a week no:-");
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        n=Integer.parseInt(br.readLine());
        switch(n)
        {
            case 1: System.out.print("Sunday");
                break;
            case 2: System.out.print("Monday");
                break;
            case 3: System.out.print("Tuesday");
                break;
            case 4: System.out.print("Wednesday");
                break;
```

```
                        case 5: System.out.print("Thruesday");
                                break;
                        case 6: System.out.print("Friday");
                                break;
                        case 7: System.out.print("Saturday");
                                break;
                        default: System.out.print("!Wrong choice");
                }
        }
        public static void main(String s[])throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P31) Write a program to enter a month no. and the print a related month name.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter a week no:-");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                switch(n)
                {
                        case 1: System.out.print("Jan");
                                break;
                        case 2: System.out.print("Feb");
```

```
                    break;
                case 3: System.out.print("mar");
                    break;
                case 4: System.out.print("Apr");
                    break;
                case 5: System.out.print("May");
                    break;
                case 6: System.out.print("Jun");
                    break;
                case 7: System.out.print("july");
                    break;
                case 8: System.out.print("Aug");
                    break;
                case 9: System.out.print("sep");
                    break;
                case 10: System.out.print("Oct");
                     break;
                case 11: System.out.print("Nov");
                     break;
                case 12: System.out.print("Dec");
                     break;
                default: System.out.print("! Wrong Choice");
        }
}
public static void main(String s[])throws Exception
{
        atul ob=new atul();
        ob. ass();
}
```

```
}
```

P32) Write a program to create a simple calculator by using switch.

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n,a,b,c;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("1.Add\n2.Sub\n3.Mul\n4.Div\n");
                System.out.print("Enter your choice=");
                n=Integer.parseInt(br.readLine());
                System.out.print("Enter two value=");
                a=Integer.parseInt(br.readLine());
                b=Integer.parseInt(br.readLine());
                switch(n)
                {
                        case 1: c=a+b;
                                System.out.print("Sum="+c);
                                break;
                        case 2: c=a-b;
                                System.out.print("Sub="+c);
                                break;
                        case 3: c=a*b;
                                System.out.print("Mul="+c);
                                break;
                        case 4: c=a/b;
                                System.out.print("Div="+c);
                                break;
```

```
                    default: System.out.print("!Wrong Choice");
          }
     }
     public static void main(String s[])throws Exception
     {
          atul ob=new atul();
          ob. ass();
     }
}
```

Loops

In programming language, looping means to execute a set of statements for a specified number of times or till a condition remains true. In a loop, the compiler checks for the condition specified for executing the loop and continues executing the statements mentioned within the loop till the condition remains true. The execution of the loop stops as soon as the condition becomes false.

Re-execution of a statement is called loops while the condition remains true.

Types of loop

1)for loop

2)nested for loop

3)while loop

4)do while loop

1)for loop:-

Syntax:-

```
for(initialization, condition, Inc/Dec)
{
     Statements;
}
```

P33) Write a program to print your name 10 times.

```
class atul
{
     void ass()
```

```
        {
                for(int i=1;i<=10;i++)
                {
                        System.out.print("Atul Kumar Varshney\n");
                }
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P34) Write a program to print 1 to 10.

```
class atul
{
        void ass()
        {
                for(int i=1;i<=10;i++)
                {
                        System.out.println(i);
                }
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P35) Write a program to print a to z.

```
class atul
```

```
{
        void ass()
        {
                for(char i='a';i<='z';i++)
                {
                        System.out.print("\t"+i);
                }
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P36) Write a program to print A to Z.

```
class atul
{
        void ass()
        {
                for(char i='a';i<='z';i++)
                {
                        System.out.print("\t"+i);
                }
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P36) Write a program to print A to Z.

```
class atul
{
        void ass()
        {
                for(char i='A';i<='Z';i++)
                {
                        System.out.print("\t"+i);
                }
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P37) Write a program to print a Fibonacci sequence.

```
class atul
{
        void ass()throws Exception
        {
                int a=0,b=1,n,c;
                System.out.print("Enter the limit=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                if(n>2)
                {
                        System.out.print("%d\t%d\t",a,b);
                        for(int i=3;i<=n;i++)
                        {
```

```
                                        c=a+b;
                                        System.out.print("%d\t",c);
                                        a=b;
                                        b=c;
                                }
                        }
                        else
                        {
                                System.out.print("The value is grater the 2");
                        }
                }
                public static void main(String s[])throws Exception
                {
                        atul ob=new atul();
                        ob. ass();
                }
        }
```

2)<u>**Nested for Loop**</u>**:-**

Syntax:-

```
                for(initialization,condition,Inc/Dec)
                {
                        for(initialization,condition,Inc/Dec)
                        {
                                Statements;
                        }
                        for(initialization,condition,Inc/Dec)
                        {
                                Statements;
                        }
```

```
        }
```

P38) Write a program to print the pattern

```
*
**
***
****
*****
******
```

```
import java.io.*;
class atul
{
    void ass()throws Exception
    {
        int n;
        System.out.print ("Enter the limit=");
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        n=Integer.parseInt(br.readLine());
        for(int i=1;i<=n;i++)
        {
            for(int j=1;j<=i;j++)
            {
                System.out.print("*");
            }
            System.out.print("\n");
        }
    }
    public static void main(String s[]) throws Exception
    {
        atul ob=new atul();
```

```
                ob. ass();
        }
}
```

P39) Write a program to print the pattern

```
1
12
123
1234
12345
123456
```

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter the limit=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=i;j++)
                        {
                                System.out.print(j);
                        }
                        System.out.print("\n");
                }
        }
        public static void main(String s[]) throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P40) Write a program to print the pattern

```
    *
   * *
  ***
 ****
*****
```

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter the limit=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=(n-i);j++)
                        {
                                System.out.print(" ");
                        }
                        for(int k=1;k<=i;k++)
                        {
                                System.out.print("*");
                        }
                        System.out.print("\n");
                }
        }
        public static void main(String s[]) throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P41) Write a program to print the pattern

```
     1
    12
   123
  1234
 12345
123456
```

```
import java.io.*;
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter the limit=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=(n-i);j++)
                        {
                                System.out.print(" ");
                        }
```

```
                    for(int k=1;k<=i;k++)
                    {
                              System.out.print("%d",k);
                    }
                    System.out.print("\n");
          }
    }
    public static void main(String s[]) throws Exception
    {
          atul ob=new atul();
          ob. ass();
    }
}
```

P42) Write a program to print the pattern

```
*          *
**        **
***      ***
****    ****
*****  *****
************
```

```
import java.io.*;
class atul
{
    void ass()throws Exception
    {
          int n;
          System.out.print("Enter the limit=");
          BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
          n=Integer.parseInt(br.readLine());
          for(int i=1;i<=n;i++)
```

```
                {
                        for(int j=1;j<=i;j++)
                        {
                                System.out.print("*");
                        }
                        for(int l=1;l<=(n*2)-(i*2);l++)
                        {
                                System.out.print(" ");
                        }
                        for(int k=1;k<=i;k++)
                        {
                                System.out.print("*");
                        }
                        System.out.print("\n");
                }
        }
        public static void main(String s[]) throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
}
```

P43) Write a program to print the pattern

```
1          1
12        21
123      321
1234    4321
12345  54321
123456654321
```

```
import java.io.*;
```

```
class atul
{
        void ass()throws Exception
        {
                int n;
                System.out.print("Enter the limit=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=i;j++)
                        {
                                System.out.print(j);
                        }
                        for(int l=1;l<=(n*2)-(i*2);l++)
                        {
                                System.out.print(" ");
                        }
                        for(int k=i;k>=1;k--)
                        {
                                System.out.print(k);
                        }
                        System.out.print("\n");
                }
        }
        public static void main(String s[]) throws Exception
        {
                atul ob=new atul();
                ob. ass();
        }
```

}

P44) Write a program to print the pattern

```
************
*****    *****
****       ****
***          ***
**             **
*                *
```

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int n;
                System.out.print("Enter the limit=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=(n+1)-i;j++)
                        {
                                System.out.print("*");
                        }
                        for(int l=1;l<=a;l++)
                        {
                                System.out.print(" ");
                        }
                        a=a+2;
                        for(int k=1;k<=(n+1)-i;k++)
                        {
                                System.out.print("*");
                        }
                        System.out.print("\n");
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P45) Write a program to print

```
*
***
*****
*******
*********
*******
*****
***
*
```

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                for(int i=1;i<=5;i++)
                {
                        for(int j=1;j<=i;j++)
                        {
                                System.out.print("*");
                        }
                        System.out.print("\n");
                }
                for(int i=1;i<=4;i++)
                {
                        for(int j=1;j<=(5-i);j++)
                        {
                                System.out.print("*");
                        }
                        System.out.print("\n");
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

3) <u>while loop</u>:-

<u>Syntax</u>:-

```
        while(condition)
        {
                Statements;
        }
```

P46) Write a program to print your name 10 times.

```
import java.io.*;
class atul
{
		void ass( ) throws Exception
		{
			int i=1;
			while(i<=10)
			{
				System.out.print("Atul\n");
				i++;
			}
		}
		public static void main(String s[ ]) throws Exception
		{
			atul ob=new atul( );
			ob. ass( );
		}
}
```

P47) Write a program to print the reverse of any value.

```
import java.io.*;
class atul
{
		void ass( ) throws Exception
		{
			int n,rev=0,r;
			System.out.print("Enter any value=");
			BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
			n=Integer.parseInt(br.readLine());
			while(n!=0)
			{
				r=n%10;
				rev=(rev*10)+r;
				n=n/10;
```

```
                }
                System.out.print("Reverse of any value="+rev);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P48) Write a program to check that the given value is palindrome or not.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int n,t,r,rev=0;
                System.out.print("Enter any value=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                t=n;
                while(n!=0)
                {
                        r=n%10;
                        rev=(rev*10)+r;
                        n=n/10;
                }
                if(rev==t)
                {
                        System.out.print("Value is palindrome");
                }
                else
                {
                        System.out.print("Value is not palindrome");
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P49) Write a program to check that the given value is Armstrong or not.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int n,r,t,arm=0;
                System.out.print("Enter any value=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                n=Integer.parseInt(br.readLine());
                t=n;
                while(n!=0)
                {
                        r=n%10;
                        arm=(r*r*r)+arm;
                        n=n/10;
                }
                if(arm==t)
                {
```

```
                                System.out.print("Value is Armstrong");
                    }
                    else
                    {
                                System.out.print("Value is not Armstrong");
                    }
          }
          public static void main(String s[ ]) throws Exception
          {
                    atul ob=new atul( );
                    ob. ass( );
          }
}
```

4)<u>do while</u>:-

<u>Syntax</u>:-

```
do
{
                    Statements;
}while(condition);
```

Difference between While and do while loop.

While	**Do while**
In While loop the controlling condition appears at start of the loop.	I do while loop the controlling condition appears at end of the loop.
The statements do not execute if the condition is false.	The statements is execute at least one time if the condition is true as well as false.
Entry controlled loop.	Exit controlled loop.

P50) Write a program to create a simple calculator in re-execution mode if we press y.

```
import java.io.*;
class atul
{
          void ass( ) throws Exception
```

```
	{
		int a,b,c,n;
		char ch='n';
		do
		{
			System.out.print("1.Add\n2.Sub\n3.Mul\n4.Div\n");
			System.out.print("Enter your choice=");
			BufferedReader		br=new		BufferedReader(new
InputStreamReader(System.in));
			n=Integer.parseInt(br.readLine());
			System.out.print("Enter two values=");
			a=Integer.parseInt(br.readLine());
			b=Integer.parseInt(br.readLine());
			switch(n)
			{
				case 1: c=a+b;
					System.out.print("Sum="+c);
					break;
				case 2: c=a-b;
					System.out.print("Sub="+c);
					break;
				case 3: c=a*b;
					System.out.print("Mul="+c);
					break;
				case 4: c=a/b;
					System.out.print("Div="+c);
					break;
				default: System.out.print("!Wrong Choice");
			}
			System.out.print("\nDo you want to cont press y=");
```

```
                ch=(char)br.read();
            }while(ch=='y');

        }
        public static void main(String s[ ]) throws Exception
        {
            atul ob=new atul( );
            ob. ass( );
        }
}
```

Break Statement

The break statement is one of the loop control statement that terminates the loop. C programming language uses the keyword 'break' for using the break statement. In other words, whenever the compiler encounters the break statement it terminates the loop.

P51) W.A.P. to show the break statement.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
            int n;
            System.out.print("Enter the value to break a loop:-");
            BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
            n=Integer.parseInt(br.readLine());
            for(int i=1;i<=100;i++)
            {
                if(i==n)
                {
                    break;
                }
```

```
                    System.out.print("Atul\n"+i);
            }
        }
        public static void main(String s[ ]) throws Exception
        {
            atul ob=new atul( );
            ob. ass( );
        }
}
```

Continue Statement

The continue statement is the next loop control statement that continues to execute the loop. C programming language use the keyword ‘continue’ for using the continue statement. In other words, whenever the continue statement arrives the compiler immediately skips the current statements of the loop and starts the next iteration of the loop.

P52) W.A.P. to show the continue statement it exc. the 2 value after printing the value.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
            for(int i=1;i<=10;i++)
            {
                    if(i==2)
                    {
                            continue;
                    }
                    System.out.print("\n"+i);
            }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

Array

An array is a type od data structure that can hold multiple values of similar type of data. According to the technical definition of the array, “Array is a homogeneous data structure that sequentially stores the same types of operations on these data”.

Array can store similar data type. Each data item of an array is called the array element. For Ex:- if the data type of the array is char then it only stores the character data or string.

Types of an Array

1)1-D Array

2)2-D Array

1)<u>1-D Array</u>:-

<u>Array Declaration</u>:-

<u>Syntax</u>:-

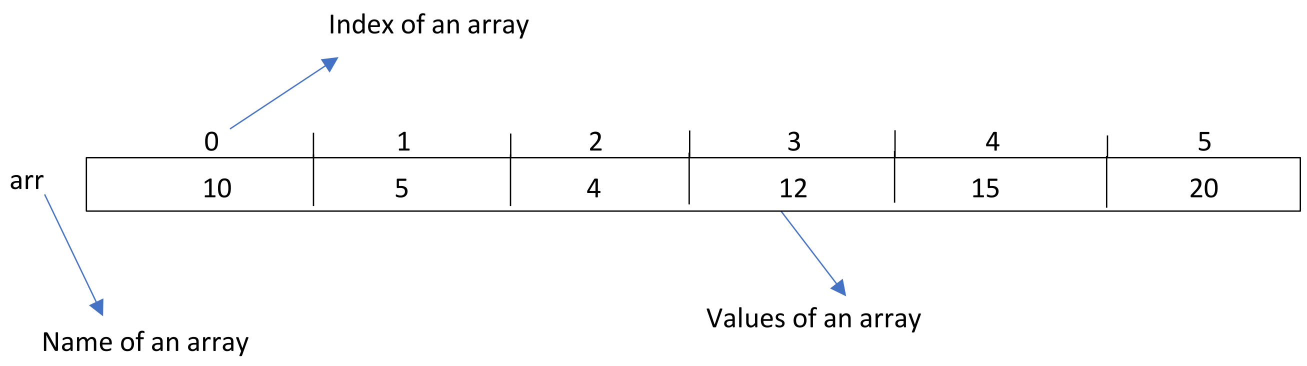
Data_type name_of_array[]=new Data_type [Size];

Ex:- int arr[]=new int[6];

Declares an integer array where the size of the array is 6. The name of the array is arr. It can store any 6 numbers that would be stored from location 0 to 6 in the computer memory.

The index no. of an array start from 0 to size-1.

An array looks like.



<u>Initialisation at Compile Time</u>:-

int arr[6]={12,13,5,3,1,15};

Initialisation at Run Time:-

```
        int arr[]=new int[6];
        System.out.print("Enter values in an array=");
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        for(int i=0;i<6;i++)
        {
                arr[i]=Integer.parseInt(br.readLine());
        }
```

Accessing Array Elements:-

```
        System.out.print(arr[2]);
```

P54) W.A.P. to sum of all elements in an 1-D array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int s=0;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                for(int i=0;i<6;i++)
                {
                        s=s+arr[i];
                }
                System.out.print("Sum of all elements="+s);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P55) W.A.P. to find maximum value in a 1 D Array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int max;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                max=arr[0];
                for(int i=0;i<6;i++)
                {
                if(max<arr[i])
                {
                        max=arr[i];
                }
                System.out.print("Maximum value in an array="+max);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P56) W.A.P. to find minimum value in a 1-D Array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int min;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                min=arr[0];
                for(int i=0;i<6;i++)
                {
                        if(min>arr[i])
                        {
                                min=arr[i];
                        }
                }
                System.out.print("Minimum value in an array="+min);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P57) W.A.P. to find any element in a 1-D Array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int n,f=0;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                System.out.print("What value you will find=");
                n=Integer.parseInt(br.readLine());
                for(int i=0;i<6;i++)
                {
                        if(n==arr[i])
                        {
                                f=1;
                        }
                }
                if(f==1)
                {
                        System.out.print("Value present in an array");
                }
                else
                {
                        System.out.print("Value not present in an array");
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P58) W.A.P. to short an array in Ascending Order.

```
 import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int t;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                for(int i=0;i<6;i++)
                {
                        for(int j=i;j<6;j++)
                        {
                                if(arr[i]>arr[j])
                                {
                                        t=arr[i];
                                        arr[i]=arr[j];
                                        arr[j]=t;
                                }
                        }
                }
                System.out.print("The shorted array=");
                for(int i=0;i<6;i++)
                {
                        System.out.print("\t"+arr[i]);
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P59) W.A.P. to short an array in descending order.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int t;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                for(int i=0;i<6;i++)
                {
                        for(int j=i;j<6;j++)
                        {
                                if(arr[i]<arr[j])
                                {
                                        t=arr[i];
                                        arr[i]=arr[j];
                                        arr[j]=t;
                                }
                        }
                }
                System.out.print("The Shorted array=");
                for(int i=0;i<6;i++)
                {
                        System.out.print ("\t"+arr[i]);
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P60) W.A.P. to replace a particular index value in 1-D array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int n,p;
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                System.out.print("Values in an array=");
                for(int i=0;i<6;i++)
                {
                        System.out.print("\t"+arr[i]);
                }
                System.out.print("\nWhat value you want to replace=");
                n= Integer.parseInt(br.readLine());
                System.out.print("Enter the index of a value=");
                p= Integer.parseInt(br.readLine());
                arr[p]=n;
                System.out.print("The change array=");
                for(int i=0;i<6;i++)
                {
                        System.out.print("\t"+arr[i]);
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P61) W.A.P. to replace value if the array element is even replace 0 if the array element is odd replace 1.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                for(int i=0;i<6;i++)
                {
                        if(arr[i]%2==0)
                        {
                                arr[i]=0;
                        }
                        else
                        {
                                arr[i]=1;
                        }
                }
                System.out.print("The Change array=");
                for(int i=0;i<6;i++)
                {
                        System.out.print("\t"+arr[i]);
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P62) W.A.P. to Add two array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[]=new int[6];
                int arr1[]=new int[6];
                int sum[]=new int[6];
                System.out.print("Enter values in an array=");
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                System.out.print("Enter values in second array=");
                for(int i=0;i<6;i++)
                {
                        arr[i]= Integer.parseInt(br.readLine());
                }
                for(int i=0;i<6;i++)
                {
                        sum[i]=arr[i]+arr[i];
                }
                System.out.print("The add array=");
                for(int i=0;i<7;i++)
                {
                        System.out.print("\t"+sum[i]);
                }
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

2)<u>2-D Array</u>:-

If the array contains two dimensions then it is called the two- dimensional array or 2-D array. This 2D-array is also called the matrix. Matrix is a combination of the row and columns.

<u>2-D Array Declaration:-</u>

<u>Syntax</u>:- data_type name_of array[][]=new data_type[row][columns];

<u>Ex</u>:- int arr[][]=new int [3][4];

The following statement declares and initializes an integer 2-D array where the row size and column size of the array are 3 and 4 respectively.

<u>An 2-D array looks like</u>:-

Initialization at Compile Time:-

```
int arr[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};
```

Initialization at Run Time:-

```
int arr[][]=new int[3][4];
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
System.out.print("Enter values in an array=");
for(int i=0;i<3;i++)
{
        for(int j=0;j<4;j++)
        {
                arr[i][j]=Integer.parseInt(br.readLine());
        }
}
```

Accessing array Elements:-

```
System.out.print(arr[2][2]);
```

P63) W.A.P. to sum of all values in 2-D array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
```

```
                    }
                }
                for(int i=0;i<3;i++)
                {
                    for(int j=0;j<3;j++)
                    {
                        s=s+arr[i][j];
                    }
                }
                System.out.print("Sum of all values="+s);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P64)W.A.P. to find maximum no. in 2-D array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int max;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                    for(int j=0;j<3;j++)
                {
                        arr[i][j]=Integer.parseInt(br.readLine());
                }
        }
        max=arr[0][0];
        for(int i=0;i<3;i++)
        {
                for(int j=0;j<3;j++)
                {
                        if(max<arr[i][j])
                        {
                                max=arr[i][j];
                        }
                }
        }
        System.out.print("Maximum value in an array=%d",max);
    }
    public static void main(String s[ ]) throws Exception
    {
        atul ob=new atul( );
        ob. ass( );
    }
}
```

P65) W.A.P. to find minimum value in 2-D array.

```
import java.io.*;
class atul
{
    void ass( ) throws Exception
    {
        int arr[][]=new int[3][3];
```

```
                int min;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print(“Enter values in an array=”);
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                min=arr[0][0];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(min>arr[i][j])
                                {
                                        min=arr[i][j];
                                }
                        }
                }
                System.out.print("Minimum value in an array="+min);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P66) W.A.P. to add all diagonal element in 2-D array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int s=0;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print(“Enter values in an array=”);
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i==j)
                                {
                                        s=s+arr[i][j];
                                }
                        }
                }
                System.out.print("Add all diagonal="+s);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P67)W.A.P. to find maximum in diagonal in 2-D array.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int max;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                max=arr[0][0];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i==j)
                                {
                                        if(max<arr[i][j])
                                        {
                                                max=arr[i][j];
                                        }
                                }
                        }
                }
                System.out.print("Maximum value in diagonal="+max);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P68)W.A.P. to find minimum value in diagonal.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int min;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                min=arr[0][0];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i==j)
                                {
                                        if(min>arr[i][j])
                                        {
                                                min=arr[i][j];
                                        }
                                }
                        }
                }
                System.out.print("Minimum value in an array="+min);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P69)W.A.P. to find maximum value in lower triangular.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int max;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                max=arr[1][0];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i>j)
                                {
                                        if(max<arr[i][j])
                                        {
                                                max=arr[i][j];
                                        }
                                }
                        }
                }
                System.out.print("Maximum in lower triangular="+max);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P70) W.A.P. to find minimum in value in lower triangular.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int min;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                min=arr[1][0];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i>j)
                                {
                                        if(min>arr[i][j])
                                        {
                                                min=arr[i][j];
                                        }
                        }
                    }
                }
                System.out.print("Minimum value in an lower triangular="+min);
            }
            public static void main(String s[ ]) throws Exception
            {
                atul ob=new atul( );
                ob. ass( );
            }
}
```

P71) W.A.P. to find maximum value in upper triangular.

```
import java.io.*;
class atul
{
            void ass( ) throws Exception
            {
                int arr[][]=new int[3][3];
                int max;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                    for(int j=0;j<3;j++)
                    {
                        arr[i][j]=Integer.parseInt(br.readLine());
                    }
                }
                max=arr[0][1];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i<j)
                                {
                                        if(max<arr[i][j])
                                        {
                                                max=arr[i][j];
                                        }
                                }
                        }
                }
                System.out.print("Maximum value in upper triangular="+max);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P72) W.A.P. to find minimum value in upper triangular.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int min;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                min=arr[0][1];
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i<j)
                                {
                                        if(min>arr[i][j])
                                        {
                                                min=arr[i][j];
                                        }
                                }
                        }
                }
                System.out.print("Minimum value in an upper triangular="+min);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P73) W.A.P. to add all elements in lower triangular.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int s=0;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i>j)
                                {
                                        s=s+arr[i][j];
                                }
                        }
                }
                System.out.print("Sum of all lower triangular elements=%d",s);
        }
        public static void main(String s[ ]) throws Exception
```

```
          {
                    atul ob=new atul( );
                    ob. ass( );
          }
}
```

P74) W.A.P. to add all upper triangular elements.

```
import java.io.*;
class atul
{
          void ass( ) throws Exception
          {
                    int arr[][]=new int[3][3];
                    int s=0;
                    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                    System.out.print("Enter values in an array=");
                    for(int i=0;i<3;i++)
                    {
                              for(int j=0;j<3;j++)
                              {
                                        arr[i][j]=Integer.parseInt(br.readLine());
                              }
                    }
                    for(int i=0;i<3;i++)
                    {
                              for(int j=0;j<3;j++)
                              {
                                        if(i<j)
                                        {
                                                  s=s+arr[i][j];
                                        }
                    }
                }
                System.out.print("Add all upper triangular elements="+s);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

P75) W.A.P. to add all diagonal elements.

```
import java.io.*;
class atul
{
        void ass( ) throws Exception
        {
                int arr[][]=new int[3][3];
                int s=0;
                BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Enter values in an array=");
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                arr[i][j]=Integer.parseInt(br.readLine());
                        }
                }
                for(int i=0;i<3;i++)
                {
                        for(int j=0;j<3;j++)
                        {
                                if(i==j)
                                {
                                        s=s+arr[i][j];
                                }
                        }
                }
                System.out.print("Add all diagonal elements="+s);
        }
        public static void main(String s[ ]) throws Exception
        {
                atul ob=new atul( );
                ob. ass( );
        }
}
```

Method in Java

There are two type of method used in java.

1)non-parameterize

2) Parameterize

1)non-parameterize

```
Ex:- class atul
        {
                void add()
                {
                        Int a=10,b=20,c;
                        c=a+b;
                        System.out.print("Add="+c);
                }
                public static void main(String s[])
                {
                        atul ob=new atul();
                        ob.add();
                }
        }
```

2) Parameterize:-

```
Ex:-  class atul
        {
                void add(int a,int b)
                {
                        int c;
                        c=a+b;
                        System.out.print("Add="+c);
                }
                public static void main(String s[])
                {
                        atul ob=new atul();
                        ob.add(10,20);
                }
        }
```

Constructor in Java

In Java, a constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

Rules for Creating a Constructor

1) Class name and constructor name must be same.
2) Constructor don't have any return type.
3) Constructor can't be static,final,abstract.
4) Constructor don't call constructor call it self.

Types of Constructor

1) Default constructor (no-arg constructor)
2) Parameterized constructor

1) Default constructor (no-arg constructor):-

```
class atul

{

        atul()

        {

                System.out.print("Hello I am Atul");

        }

        public static void main(String a[])

        {

                atul ob=new atul();

        }

}
```

2) Parameterized constructor

```
class atul

{
        atul(int a,int b)
        {
                int c;
                c=a+b;
                System.out.print("Add="+c);
        }
        public static void main(String s[])
        {
                atul ob=new atul(10,20);
        }
}
```

Static

Static Variable

If you declare any variable as static, it is known as a static variable.

The static variable gets memory only once in the class area at the time of class loading.

Ex:-

```
class atul
{
        int a=0;
        atul()
        {
                a++;//incrementing value
                System.out.println(a);
        }
        public static void main(String args[])
        {
                atul c1=new atul();
                atul  c2=new atul();
                atul  c3=new atul();
        }
}
```

If we can't use variable as static then it is can't common for all objects. It will get a memory every time.

Ex:-

```
class atul
{
        static int a=0;
        atul()
        {
                a++;//incrementing value
                System.out.println(a);
        }
        public static void main(String args[])
        {
                atul c1=new atul();
                atul  c2=new atul();
                atul  c3=new atul();
        }
}
```

If we can use variable as static then it is common for all objects. It will get a memory one time.

Static Method

- A static method belongs to the class rather than the object of a class.
- A static method can be invoked without the need for creating an instance of a class.
- A static method can access static data member and can change the value of it.

Ex:-

```
class atul
{
        static int a=10;
        static void as()
        {
                System.out.print(a);
        }
        public static void main(String s[])
```

```
          {
                    atul ob=new atul();
                    ob.as();
          }
}
```

Static method only excess the static data member. If we can excess the non static data member in static method then it can shows an error.

<u>Static Block</u>

*After loading the class the compiler excess the static block and then main function.

Ex:-

```
class atul
{
          static
          {
                    System.out.print("Hello");
          }
}
```

If you are using JDK 1.7 OR MORE. Then it can show an error.

JDK 1.7 after loading the class the it can find static block and then main function.

this keyword

1. this can be used to refer current class instance variable.
2. this can be used to invoke current class method (implicitly)
3. this() can be used to invoke current class constructor.
4. this can be passed as an argument in the method call.
5. this can be passed as argument in the constructor call.

6. this can be used to return the current class instance from the method.

Ex:-

```
class atul
{
    int a;
    atul(int a)
    {
            a=a;
    }
    void display()
    {
            System.out.println(a);
    }
    public static void main(String args[])
    {
            Student s1=new Student(10);
            Student s2=new Student(20);
            s1.display();
            s2.display();
    }
}
```

it can display 0 because it can’t consider it is an local variable or Global.

1)Current class instance variable

```
class atul
{
    int a;
    atul(int a)
    {
            this.a=a;
    }
    void display()
    {
            System.out.println(a);
```

```
        }
        public static void main(String args[])
        {
                Student s1=new Student(10);
                Student s2=new Student(20);
                s1.display();
                s2.display();
        }
    }
```

It display 10 20 because this can be used to refer current class instance variable.

2)Current class method

```
class atul

{

        void as()

        {

                s.o.p("Hello");

        }

        void ps()

        {

                s.o.p("hi");

                this.as();

        }

        psvm()

        {

                atul ob=new atul();
                ob.ps();
        }
}
```

3) Current Class constructor

```
class atul
{
    atul()
    {
        s.o.p("Hello");
    }
    atul(int a)
    {
        s.o.p("hi");
        this();
    }
    p.s.v.m()
    {
        atul ob=new atul(20);
    }
}
```

4) Pass as ab argument in the method

```
class atul
```

```
{
        void as(atul obj)
        {
                s.o.p("Hello");
        }
        ar()
        {
                as(this);
        }
        p.s.v.m()
        {
                atul ob=new atul();
                ob.ps();
        }

}
```

5) Pass arguments as a constructor call

```
class  atul
{
    int a=10;
    atul()
    {
    }
}
```

6) Return current class instance

```
class atul
{
    atul as()
    {
            return this;
    }
    void ps()
    {
            s.o.p("Hello");
    }
    ps.v.m()
    {
            new atul().as().ps();
    }

}
```

Inheritance

Inheritance is use to inherit the properties of base class in to child Class.

Type of Inheritance

1)single

2)Multi Level

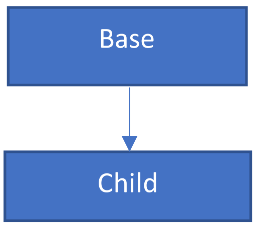
3) Hierarchical

4)Multiple

5)Hybrid

1)single :-

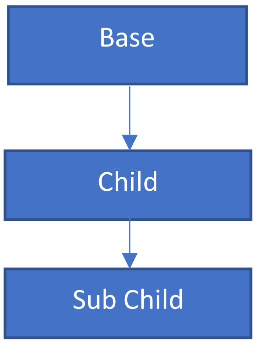
A single base class inherit by the single child class .



Ex:-

```
class base
{
        void as()
        {
                System.out.print("Hello");
        }
}
class                   child                   extends                 base
{
        void ps()
        {
                System.out.print("Hi");
        }
        public static void main(String s[])
        {
                child c=new child();
                c.as();
                c.ps();
        }
}
```

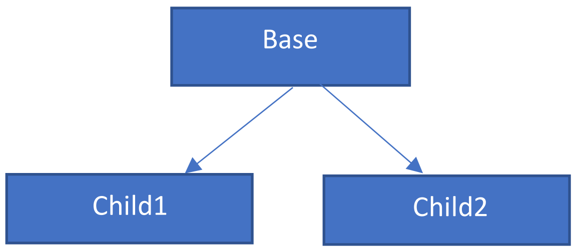
2)<u>Multi Level</u>



Ex:-

```
class base
{
            void as()
            {
                System.out.print("Hello");
            }
}
class child extends base
{
            void ps()
            {
                System.out.print("HI");
            }
}
class Subchild extends child
{
            void ds()
            {
                System.out.print("HI");
            }
            public static void main(String s[])
            {
                Subchild ob=new Subchild();
                ob.as();
                ob.ps();
                ob.ds();
            }
}
```

3) Hierarchical



```
class base
{
```

```
            void as()
            {
                System.out.print(“Hello”);
            }
}
class child1 extends base
{
            void ps()
            {
                System.out.print(“HI”);
            }
}
class child2 extends child1
{
            void ds()
            {
                System.out.print(“HI”);
            }
            public static void main(String s[])
            {
                child1 ob=new child1();
                ob.as();
                ob.ps();
                child2 ob=new child2();
                obj.as();
                obj.ds();
            }
}
```

4)Multiple 5)Hybrid

These two inheritance we can't access because two class can't inherit at a time .

Method Overloading

If a class has multiple methods having same name but different in parameters, it is known as **Method Overloading.**

Ex:-

```
class atul
{
        void add()
        {
                int a=10,b=20,c;
                c=a+b;
                System.out.print("Add="+c);
        }
        void add(int p,int q)
        {
                int r=p+q;
                System.out.print("Addition="+r);
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob.add();
                ob.add(10,20);
        }
}
```

Method Overriding

- Method overriding is used to provide the specific implementation of a method which is already provided by its superclass.
- Method overriding is used for runtime polymorphism
- The method must have the same name as in the parent class
- The method must have the same parameter as in the parent class.

Ex:-

```
class atul
{
        void as()
        {
                System.out.print("Hello");
        }
}
class atul1 extends atul
{
        void as()
        {
                System.out.print("Hi");
        }
        Public static void main(String s[])
        {
                atul1 ob =new atul1();
                ob.as();
        }
    }
```

Super keyword

The super keyword can also be used to invoke the parent class constructor , method.

1)current Class constructor

```
class base
{
        base()
        {
                System.out.println("Hello");
        }
}
class child extends base
{
        child()
        {
                super();
                System.out.println("hi");
        }
        public static void main(String args[])
        {
                child ob=new child();
        }
    }
```

2) parent class variable

```
class atul
{
        int a=10;
}
class atul1 extends atul
{
        int a=20;
        void as()
        {
                s.o.p(a);
                s.o.p(super.a);
        }
        p.sv.m()
        {
                atul1 ob=new atrul();
                ob.as();
        {
}
```

3) to call a parent class method

```
class atul
{
        void as()
        {
                s.o.p("Hello");
        }
}
class atul1 extends atul
{
        void ps()
        {
                s.o.p("Hi");

        }
        void da()
        {
                s.o.p("Atul");
                super.as();
                ps();
        }
        ps.v.m()
        {
                atul1 ob=new atul1();
                ob.da();
        }
}
```

Instance Initializer Block

Instance Initializer block is used to initialize the instance data member. It run each time when object of the class is created.

Ex:- class atul

```
        {
        atul()
        {
                s.o.p("Atul");
        }
        {
                s.o.p("Hello");
        }
        p.s.v.m();
        {
                atul ob=new atul();
        }
}
```

Ex;-

```
class atul
{
        int a;
        atul()
        {

                a=10;
                s.o.p(a);
        }
        {
                a=20;
                s.o.p(a);
        }
        p.s.v.m();
        {
                atul ob=new atul();
        }
}
```

Ex:-

```
class atul
{
        atul()
        {
                s.o.p("Hello");
        }
}
class atul1 extends atul
{
        atul1()
        {
                super();
                s.o.p("Hi");
        }
        atul1(int a)
        {
                super();
                s.o.p("atul");
        }
        {
                s.o.p("I am hand");
        }
        p.s.v.m()
        {
                atul1 ob=new atul1();
                atul1 obj=new atul1(10);

        }
}
```

Aggrigation In Java

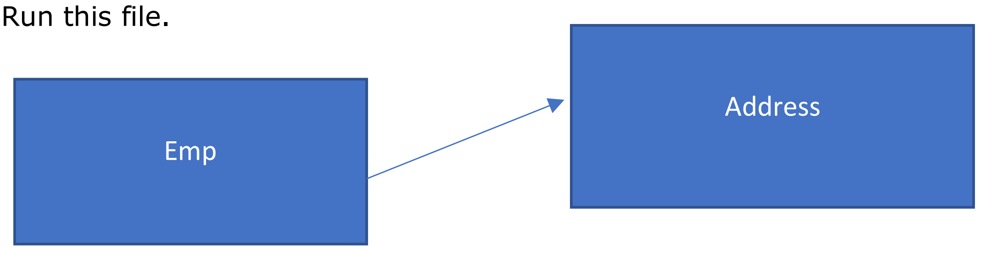
If a class have an entity reference, it is known as Aggregation

```
public class Address
{
        String city,state,country;

        public Address(String city, String state, String country)
        {
                super();
                this.city = city;
                this.state = state;
                this.country = country;
        }

}
```

Save :- Address.java

```
public class Emp
{
        int id;
        String name;
        Address address;
        public Emp(int id, String name,Address address)
        {
                this.id = id;
                this.name = name;
                this.address=address;
        }
        void display()
        {
                System.out.println(id+" "+name);
                System.out.println(address.city+" "+address.state+" "+address.country);
        }
        public static void main(String[] args)
        {
                Address address1=new Address("gzb","UP","india");
        Address address2=new Address("gno","UP","india");
        Emp e=new Emp(111,"varun",address1);
        Emp e2=new Emp(112,"arun",address2);
        e.display();
        e2.display();
        }
}
```

Save :- Emp.java
Run this file.



Class emp is use the class address funcanilaty.

Covariant Return Type

The covariant return type specifies that the return type may vary in the same direction as the subclass.
override method by changing the return type if subclass overrides any method
class atul

```
{
	atul get()
	{
		return.this;
	}
}
class atul1 extends atul
{
	atul1 get()
	{
		return.this;
	}
	void ms()
	{
		s.o.p("Hello");
	}
	p.sv.m()
	{
		new atul1().get().ms();
	}
}
```

Final Keyword

1. variable
2. method
3. class

1)final variable :-

Final variable value can’t be change. It value is final.

```
class atul

{

	final a=10;

	void as()

	{

		a++;		//it shown an error final value con’t be change
		System.out.print(a);
	}
	public static void main(String s[])
	{
		atul ob=new atul();
		ob.as();
	}
}
```

2)<u>**final Mehtod**</u>**:-**

Final method can't be override.

```
class atul
{
	final void as()
	{
		System.out.print("Hello");
	}
}
class atul1 extends atul
{
	void as()
	{
		System.out.print("Hi");
```

```
        }
        public static void main(String s[])
        {
                atul1 ob=new atul1();
                ob.as();
        }
}
```

3)<u>final Class</u>:-

Final class can’t be inherited.

```
final class base
{
        void as()
        {
                System.out.print(“Hello”);
        }
}
class child extends base
{
        void ps()
        {
                System.out.print(“Hi”);
        }
        public static void main(String s[])
        {
                Child ob=new child();
                ob.ps();
        }
}
```

Upcasting

If we can create the reference variable of base class hold by the child class is called Upcasting.

```
class base
{
        void as()
        {
                System.out.print("Hello");
        }
}
Class child extends base
{
        void ps()
        {
                System.out.print("Hi");
        }
        public static void main(String s[])
        {
                base ob=new child();
                ob.ps();
                ob.as();
        }
}
```

Dynamic binding

Wraping with data member and member function with in a single class is called dynamic binding.

```
class atul
{
        int a=10;
        void as()
        {
                System.out.print("Hello");
        }
        public static void main(String s[])
        {
                atul ob=new atul();
                ob.as();
        }
}
```

instanceOf

instance of operator is used to test whether the object is an instance of the specified type.

Ex:-

```
class atul
{
        void as()
        {
                ..............
        }
        p.s.v.m()
        {
                atul ob=new atul();
                s.o.p(ob  intanceof   atul);
        }
}
```

Output:- True

Ex:-

```
class atul
{
        atul()
        {
        }
        atul(int a)
        {
        }
}
class atul1 extends atul
{
        atul1(int a)

        {

        }

        p.s.v.m()

        {

                atul ob=new atul1();    upcasting

                s.o.p(ob intanceof atul);

        }

}
```

Abstract Class

- An abstract class must be declared with an abstract keyword.
- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have constructors and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.

```
abstract class atul

{

        abstract  void as();

        void ps()

        {

                System.out.print("Hello");

        }

}

class atul1 extends atul

{
        void as()
        {
                System.out.print("Hi");
        }
        public static void main(String s[])
        {
                atul1 ob=new atul1();
                ob.as();
                ob.ps();
        }
}
```

Interface

An **interface in Java** is a blueprint of a class. It has static constants and abstract methods.

The interface in Java is *a mechanism to achieve* abstraction. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java.

Interface don't have a normal methods.

We can't create the object of interface.

```
interface my
{
        void as();
        void ps();
}
class atul implements my
```

```
{
        void as()
        {
                System.out.print(“Hello”);
        }
        Void ps()
        {
                System.out.print(“Hi”);
        }
        public static void main(String a[])
        {
                atul ob=new atul();
                ob.as();
                ob.ps()
        }
}
```

multiple inheritance in Java by using interface because we can implement two interface at a time.

```
interface my
{
        void as();
}
interface my1
{
        void ps();
}
class atul implements my,my1
{
        void as()
        {
                System.out.print("Hello");
        }
        Void ps()
        {
                System.out.print("Hi");
        }
        public static void main(String a[])
        {
                atul ob=new atul();
                ob.as();
                ob.ps()
        }
}
```

Inner Class

Ex:-

```
class Outer
{
        void as()
        {
                s.o.p("Hello");
        }
        class Inner
        {
                void ps()
                {
                        s.o.p("Hi")
                }
        }

        p.s.v.m()
        {
                Outer o=new Outer();
                o.as();
                Outer.Inner I=o.new Inner();
                I.ps();
        }
}
```

Ex:-

```
class Outer
{
        void as()
        {
                s.o.p("Hello");
        }
        class Inner
        {
                void ps()
                {
                        s.o.p("Hi")
                }
                class Inner1
                {
                        void ds()
                        {
                                s.o.p("Hi")
                        }
                }
        }
        p.s.v.m()
        {

        }
}
```

Inner Interface

```
interface my
{
        void a();
        interface my1
        {
                void b();
        }
}
classatul implements my.my1
{
        void a()
        {
        }
        void b()
        {
        }
        p.s.vm()
        {
                my.my1 ob=new atul();
                ob.b();
        }
}
```

Difference b/w Interface and abstract

Abstract	Interface
Have both abstract and no abstract method.	Have only abstact methods but java 8 have default and static methods.
Don't support multiple inheritance.	Support multiple inheritance.
Abstract have all type of variable.	Static and final.
Abstraction con't provide implementation of interface.	Interface con't provide implementation of abstract class.

Access Modifier

Access Modifier	With in class	With in package	Outside package by subclass	Package outside
Private	Y	N	N	N
Default	Y	Y	N	N
Protected	Y	Y	Y	N
Public	Y	Y	Y	Y

Package

Like an java predefine packages we can create our own package.

import java.io.*;

import java.awt.*;

import java.swing.*;

like we can create our own package.

<u>Example Step By Step</u>

```
package mypack;
public class atul
{
            public void as()
            {
                    System.out.println("Hello");
            }
}
```

Create a folder mypack. Save it name as atul.java

Like :- D/:> mypack/atul.java

only Compile It.

```
import mypack.*;
class atul1
{
    public static void main(String args[])
    {
            atul ob = new atul();
            ob.as();
}
```

}
Save it :
Like :- :- D/:>atul1.java
Compile it and Run.

Sub Package

Example Step By Step

```
package mypack,mypack1;
public class atul
{
            public void as()
            {
                    System.out.println("Hello");
            }
}
```

Create a folder mypack and inside the folder create a new folder mypack and Save it name as atul.java

Like :- D/:> mypack/mypack1/atul.java

only Compile It.

```
    import mypack.mypack1.*;
    class atul1
    {
        public static void main(String args[])
        {
                atul ob = new atul();
                ob.as();
    }
}
```

Save it :
Like :- :- D/:>atul1.java
Compile it and Run.

Object Cloning

The object cloning is a way to create exact copy of an object.

```
class atul imlements Cloneable
{
        int a;
        atul(int a)
        {
                this.a=a;
        }
        public static void clone()throws Exception
        {
                return super.clone();
        }
        p.s.v.m()
        {
                try
                {
                        atul ob=new atul(10);
                        atul obj=(atul)ob.clone();
                        s.o.p(ob.a);
                        s.o.p(obj.a);
                }
                carch(Exception e)
                {
                }
        }
}
```

Math Function

import java.lang.Math.*;

Math.sin(a);

```
1) .abs(a);        //  Return the absalute value
                a=12.5;          // 12.5
                a=12.6;          //12.6
2) max(a,b);    // Max
                a=10;
                b=20;
                s.o.p(Math.max(a,b));  //20
3)min(a,b);     //min
                a=10;
                b=20;
                s.o.p(Math.min(a,b));  //10
4)round(a);     // round of
                a=10.5;      //10
                a=10.6;      //11
5)sqrt(a);       //square  of any value
                a=2;      //2*2
                a=3;      //3*3
6)cbrt(a);      //  cudbe
                a=2;  //    2*2*2
7)pow(a,b);   //  a^b
                a=2
                b=3
```

```
                  pow(a,b);    // 2^3
                  pow(b,a);   //3^2
8)signnum(a);       zero    zero
                       positive      +1
                       negatiive   -1
                  a=10;
                  signnum(a);   // +1
                  a=-10;
                  signnum(a);   //-1
9) ceil(a);              // smallest intiger value
                  10.4    //10
                  10.5   //10.
10)   copySign(a,b);     //   a=10;
                         b=-12;
                        copySign(a,b);       -10
                         copySign(b,a);         12
11) floor(a);     /// a=744.93;
                   floor(a);          //   744
                   a=-39.28;   // 4,5
                   floor(a);         //  -40
12) floorDiv(a,b);      a=25
                  b=2
                   floorDiv(a,b);    //25/2  =   -12.1;  //12
13) random();           //     0.1  to 1.0.
                   //  0.12548585414;
```

14) rint(a) //if the argument is negative or positive it return a integer value that is even.

```
a=81.68;      //82.0
a=-37.25;     //-38
a=80.5;       //80.0
a=79.5;       //80.0
```

15) hypot(a,b); //a=2

```
b=3;
     a^2+b^2
```

16)ulp(a); // if the value is

```
+/-  Double  //2^971.
+/-  Float   // 2^104.
Double a= 12.4588;   //2^971
floata= 12.4588;   //2^104
```

17)addExact(a,b);//a=10

```
b=20
addExact(a,b);   //30
```

18)subExact(a,b); // a=10

```
b=20
subExact(a,b);   //-10
```

19) mltiplyExact(a,b); //a=2

```
b=3
mltiplyExact(a,b);   //6
```

20)incrementExact(a); // a=7; //8

```
//a=-5;    //-4
```

21) decrementExact(a); //a=10;   //9

//a=-4;   //-5

22)negateExact(a);   //a=25;   //-25

23)toIntExact(a);   //a=25;   //25

//a=-25;   //-25

24)log();

25)log10();

26)log1p();

27)exp();

28)expm1();

29)sin();

30)cos();

31)tan();

32)asin();

33)acos();

34)atan();

35)sinh();

36)cosh();

37)tanh();

toDegrees();

toRadiaus();

Wrapper Class

Serialization:- We can convert private value in to object(Wapper Class).

Primitive type	Wrapper Class
boolean	Boolean

char	Characeter
byte	Byte
short	Short
int	Integer
long	Long
float	Float

Autoboxing :- convert private value in to wraper calss.

```
public class atul
{
        void as()
        {
                float  a=10.2;
                Integer b=Integer.valueOf(a);


        }
}
```

Unboxing:- convert Wraper class in to primitive.

```
public void atul
{
        void as()
        {
                Integer a=new Integer(3);
                int b=a.intValue();
        }
}
```

Strictfp Keyword

You will get the same result on every plaeform. if you will perform operations in the floating point variable it may different work on every palteform that's why we can use Strictfp keyword.

```
Strictfp class atul

Strictfp  interface my

Strictfp  void as()

Strictfp int a=10;   // not allowed

Strictfp  atul()     //    constructor not allowed
```

javadoc

to create a API.

```
package com.abc;

/**  I am atul

        I am  */

public class atul

{

        /**   Hello KR */

        public static void add()

        {

                Syste.out.print("Hello");

        }

}
```

cmd:- javadoc atul.java

Create a multiple web pages. Open indx.html

Command Line Arguments

```
class atul

{

        public static void main(String s[])

        {

                System.out.print("My name is"+s[0]);

        }

}
```

compile :- javac atul.java

Run:- java atul Amit

Output:- My name is Amit

String

Array of character is called String. Sequence of Characer.

import java.lang.string.*;

Ex- char name[]={'A','t','u','l'};

String name="Atul";

1)String

2)String Buffer

3)String Builder

1)<u>String</u>

String name="Atul";

char name[]={'A','t','u','l'};

String name=new String("Atul");

Immutable String:- Unchange

Sting name="Atul";

name.concat("Kumar");

s.o.p(name);   //Atul

....

Sting name="Atul";

name=name.concat("Kumar");

s.o.p(name);   //Atul

....

String Compare:-

1)equal:-

String s1="Atul";

String s2="Amit";

String s3="Atul";

s.o.p(s1.eqals(s2));   //false

s.o.p(s1.eqals(s3)); //true

....

2) (==)

String s1="Atul";

String s2="Amit";

String s3=new String("Atul");

String s4="Atul";

s.o.p.(s1==s2);   //false

s.o.p.(s1==s3);   //false

have same but using instance.

s.o.p.(s1==s4); //true

3) <u>Compareto()</u>:-

first String == second String //0

first String > second String //+5

first String < second String //-15

String s1="Atul"; 4

String s2="Amit"; 4

String s3="Nishant";

s.o.p.(s1.compareto(s2)); // 0

s.o.p.(s1.compareto(s3)); // -

s.o.p.(s3.compareto(s1)); // +

....

Sting add:-

String name="Atul"+"Kumar"; //Atul Kumar

String name="Atul";

String lastname="Kumar";

s.o.p(name.concat(lastname)); //Atul Kumar

String name=10+20+5+"Atul"+5+4; //35 Atul 5 4

String name="Atul"+5+4; //Atul 5 4

....

<u>Sub String:-</u>

Atul Kumar //tu

Sting name="Atul Kumar";

s.o.p.(name.substring(4)); // Kumar

s.o.p.(name.substring(0,4)); //Atul

```
s.o.p.(name.substring(2,4)); //ul
```

……………………………

StringBuffer and StringBuilder:-

A string that can be changed in known an mutabble string.

```
StringBuffer name=new StringBuffer("Hello");
name.append("Atul");
s.o.p(name);            //Hello Atul

StingBuffer name=new StingBuffer("Atul");
name.concat("Kumar");
s.o.p(name);      //Atul Kumar

StingBuffer name=new StingBuffer("Atul Varshney");
name.insert(4,"Kumar");   //Atul Kumar Varshney

StingBuffer name=new StingBuffer("Atul Kumar Varshney");
name.replace(1,4,"Amit");   //Atul Kumar Varshney
name.replace(1,3,"Amit");   //Amitl Kumar Varshney

StingBuffer name=new StingBuffer("Atul Kumar Varshney");
name.delete(4,8);   //Atul ar Varshney

StingBuffer name=new StingBuffer("Amit");
name.reverse();   //tima
```

```
class atul
{
        String name;
        atul(String name)
        {
                this.name;
        }
        public void toString()
        {
                return name;
        }
        p.s.vm()
        {
                atul ob=new atul("Atul");
                atul obj=new atul("Amit");
                S.o.p(ob);    //ob.toSting();
                S.o.p(obj);
        }
}
```

<u>Tokenizer</u>:-

Java.util.*;

It break the string if the space is accoure.

I am atul.

i

```
am
atul.
java.util.*;
public class atul
{
        void as()
        {
                StringTokenizer name=new  Tokenizer("I am atul");
                while(name.hasTokenizer())
                {
                        s.o.p(name.nextToken());
                }
        }
        p.s.pvm()
        {

        }
}
I
am
atul.
```

<u>String Reverse:-</u>

name.reverse();

without using function.

StringBuilder name=new StringBuilder(“Atul”);

```
For(int i=name.length()-1;i>=0;i--)
{
        S.o.p(name[i]);
}
String name="I am Atul"
```

1) charAt();

```
        s.o.p(name.charAt(2));    //
        s.o.p(name.charAt('a'));  //
```

2) concat:-

3) contains:-

```
        s.op(name.contains("atul"));     // true
        s.op(name.contains("Amit"));  //false
```

4)ends with:-

```
        s.op(name.endsWith("atul"));  //true
        s.op(name.endsWith("I"));  //false
```

5)equalsIgnoreCase():-

```
        String name="atul";
        String name1="ATUL";
        s.o.p(name.equalsIgnoreCase(name1));  //true case are ignored.
```

6)

```
String name="atul";
byte name1[]=name.getbytes();
```

```
for(int i=0;i<name1.length;i++)
{
        s.op(name1[i]);
}
```

7) getChars():-

```
        String name =new String("Bank Colony");
        char ch[ ]=new char[12];
        name.getChars(3,10,ch,0);


        Bank Colony
```

8)indexOf();

```
        String name="Atul Kumar";
        so.p(name.indexOf("u"));  //3
        so.p(name.indexOf("u",3));  //
```

9)intern():-

```
        String name="Atul";
        String name1=name.intern();
```

10)isEmpty():-

```
        String name=" ";
        s.o.p(name.isEmpty());  //
```

11) join():-

```
        String name=String.join("@","Atul","Amit","Jitu");
        s.op(name);   //Atu@Amit@Jitu
```

12)lastIndexOf:-

```
        Sring name="I am atul";
        name.laseIndexOf("a");//6
        name.lastIndexOf("a",5);        //3
```

13) length():-

14)replace():-

15)split:-

```
        String name="I am Atul";
        String name1=name.split("\s");
        for(String a:name1)
        {
                S.O.p.(a);
        }
```

Atul->   C-lang

Amit-> C++

```
I
am
atul
```

16) startWitgh();

```
        String name="am';
        s.o.p(name.startWith("a"));      //true
        s.o.p(name.startWith("m",1));  //true
```

17) Convert string in to character array:-

```
        String name="Atul";
        cahr name1=name.toCharArray();
        for(int i=0;i<name1.length();i++)
        {
                s.o.p(name1[i]);
        }
```

18)toLowerCase():-

```
        String name="ATUL";
        s.op(name.toLowerCase());
```

19)toUpperCase():-

```
        String name="atul";
        s.op(name.toUpperCase());
```

20)trim():-

remove all extra spaces.

```
        String name="  I am atul";
        s.op(name.trim());
```

21) valueOf():-

```
        int a=10;
        String n=String.valueOf(a);
        S.O.p(a+30);    //1030
```

Exception Handling

Exception Handling is use to handle the run time exception. If the statement can throw the run time error then the compiler can stop the all the remining code. By using exception handling we can handle the run time exception in a code then the remaining code will execute perfectly.

try :- Exception code

catch :- handle the exception

finally :- it executed if the exception is handled or not.

throw :- throw is use to throw an exception.

Throws :- it specific that the method is may occure exception or not.

It always used in method.

```
 Statement1
Statement2  //Run time Error
Statement3
Statement4
Statement5
Statement6
try
{
        ...............
}
catch(Exception e)
{
        s.op(e);
}
```

Ex:-

```
class atul
{
        void as()
        {
                try
                {
                        int c=10/0;
                }
                catch(Exception e)
                {
                        s.op(e);
                }
                s.op.("Hello");
        }
}
```

Ex:-

```
class atul
{
        void as()
        {
                int a=10/0;
        }
        void ps()
        {
                as();
        }
        void ds()
        {
                try
                {
                        ps();
                }
                catch(Exception d)
                {
                        s.o.p(d);
                }
        }
        p.s.v.m()
        {
                atul ob=new atul();
                ob.ds();
        }
}
```

Ex:-

```
class atul
{
        void as() throws Exception
        {
                try
                {
                        throw new ArithmeticException("Amit");  //
                }
                catch(Exception e)
                {
                        s.o.p(e);
                }
                s.o.p("hello");
        }
        p.s.v.m()
        {
        }
}
```

throw is not execute the next code it terminate the program.

Multithreading

Run a muptiple thread at simintaneausly.

perform multiple operation at same time.

Life Cycle Of thread

1)New

2)Runnable

3)Running

4)Non-Runnable

5)Terminate

Creating a thread:- there are two ways to create a thread.

1)by using Thread class

2)by using Runnable interface

1)by using Thread class

Ex:-

```
class atul extends Thread
{
        public void run()
        {
                so.p("hello");
        }
        p.s.v.m()
        {
                atul ob=new atul();
                ob.start();
        }
}
```

2)<u>by using Runnable interface</u>

Ex:-

```
class atul  implements Runnable
{
        public void run()
        {
                so.p("hello");
        }
        p.s.v.m()
        {
                atul ob=new atul();
                Thread t=new Thread(ob);
                t.start();
        }
}
```

Thread scheduling:-

This is not fix the thread scheduling first run the runnable but only one thread will execute at a time.

```
class atul extends Thread
{
	public void run()
	{
		so.p("hello");
	}
	p.s.v.m()
	{
		atul ob=new atul();
		atul obj=new atul();
		ob.start();
		obj.start();
	}
}
```

Sleep Method:-

Ex:-

```
class atul extends Thread
{
	public void run()
	{
		for(int i=1;i<=10;i++)
		{
```

```
                    try
                    {
                              Thread.sleep(500);
                    }
                    catch(Exception e)
                    {
                              S.o.p(e);
                    }
                    S.o.p(i);
          }
     }
     p.s.v.m()
     {
          atul ob=new atul();
          atul obj=new atul();
          ob.start();
          obj.start();
     }
}
```

…………………………………….

We can’t start thread twise.

```
atul ob=new atul();
ob.start();
ob.start();
```

……………………………………..

<u>join:-</u> join method is use to wait the current thread is stop then it will start a next thread.

Ex:-

```
class atul extends Thread
{
        public void run()
        {
                for(int i=1;i<=10;i++)
                {
                        try
                        {
                                Thread.sleep(500);
                        }
                        catch(Exception e)
                        {
                                S.o.p(e);
                        }
                        S.o.p(i);
                }
        }
        p.s.v.m()
        {
                atul ob=new atul();
                atul obj=new atul();
                atul obj1=new atul();
                ob.start();
                try
                {
                        ob.join();
                }
                catch(Exception e)
                {
                        s.o.p(e);
                }
                obj.start();
                obj1.start();
        }
}
```

<u>How to set a thread name and get a thread name:</u>-

initialy the thread name is Thread-0

Thread-1.

Ex:-

```
class atul extends Thread
{
        public void run()
        {
                s.o.p("hello");
                s.o.p(Thread.currentThread().getName());
        }
        p.s.v.m()
        {
                atul ob=new atul();
                atul obj=new atul();
                ob.start();
                s.o.p(ob.getName());   //Thread-0
                ob.setName("Amit");

                s.o.p(ob.getName());  //Amit
                obj.start();
        }
}
```

to know what currentb thread will run:-

```
        s.o.p(Thread.currentThread().getName());
```

Priority of Thread:-

every thread have own priority that is 1 to 10. Some time it decided on that priority bt JVM.

MIN_PRIORITY(1);

NORM_PRIORITY(5);

MAX_PRIORITY(1);

Ex:-

```
class atul extends Thread
{
        public void run()
        {
                s.o.p("hello");
                s.o.p(Thread.currentThread().getName());
        }
        p.s.v.m()
        {
                atul ob=new atul();
                atul obj=new atul();
                ob.setPriority(Thread.MAX_PRIORITY);
                obj.setPriority(Thread.MIN_PRIORITY);
                ob.start();
                obj.start();
        }
}
```

Synchronization

to countrole the multi threading . allowed only one thread at a time.

Ex:-

```
class atul
{
        synchronization void as()
        {
                s.o.p("Hello");
                try
                {
                        Thread.sleep(500);
                }
                catch(Exception e)
                {
                }
        }
}
class atul1 extends Thread
{
        atul a;
        aul1(atul a)
        {
                this.a=a;
        }
        public void run()
        {
                a.as();
        }
        p.s.v.m()
        {
                atul1 ob=new atul1();
                atul1 obj=new atul1();
                ob.start();
                obj.run();
        }
}
```

First it will complete the first thread and then run the second thread.

Synchronization block:-

If you have to create some thread free code.

```
void as()

{

        ....................................

        ....................................

        ....................................

        synchronization(this)

        {

                .............................................

        }

}
```

Input / Output

We can input a file and the output (write) a file basically we can use to perform the file handling.

Output Stream:- is use to write a data from a file.

FileOutputStream , ByteOutputStream , FileOutputStream, PipedOutputStream,ObjectOutputStream

Input Stream:- It is use to read a file.

1)FileOutputStream:-

```
import java.io.*;
public class atul
 {
   public static void main(String args[])
   {
        try
        {
          FileOutputStream fout=new FileOutputStream("as.txt");
          String s="Welcome to javaTpoint.";
          byte b[]=s.getBytes();
```

```
            fout.write(b);
            fout.close();
           }
           catch(Exception e)
           {
                 System.out.println(e);
            }
      }
}
```

2)fileInptStream:-

```
import java.io.*;
public class atul
 {
    public static void main(String args[])
    {
        try
        {
          FileInputStream fin=new FileInputStream("as.txt");
          int i=0;
          while((i=fin.read())!=-1)
          {
                     System.out.print((char)i);
          }
          fin.close();
        }
        catch(Exception e)
        {
              System.out.println(e);
         }
       }
      }
```

3)BufferOutputStream:-

```
FileOutputStream ob=new FileOutputStream("as.text");
BufferedOutputStream obj=new BufferedOutputStream(ob);
String name="I am atul";
Byte n[]=s.getBytes();
obj.write(n);
obj.flush();
obj.close();
ob.close();
```

4)BufferInputStream:-

```
FileOutputStream ob=new FileOutputStream("as.text");
BufferedIputStream obj=new BufferedInputStream(ob);
int I;
while((i=obj.read())!=-1)
{
    So.p((char)i);
}
obj.close();
ob.close();
```

5)SequenceInputStream:-

```
FileOutputStream ob=new FileOutputStream("as.text");
FileOutputStream obj=new FileOutputStream("as1.text");
SequenceInputStream st=new SequenceInputStream(ob,obj);
int I;
while((i=st.read())!=-1)
{
    S.o.p((char)i);
}
```

If we can read data from more then 2 files.

```
FileOutputStream obj=new FileOutputStream("as.text");
FileOutputStream obj1=new FileOutputStream("as1.text");
FileOutputStream obj2=new FileOutputStream("as2.text");
FileOutputStream obj3=new FileOutputStream("as3.text");
Vector v=new Vector();
v.add(obj); v.add(obj1); v.add(obj2); v.add(obj3);
Enumeration e=v.elements();
int i=0;
while((i=otp.read())!=-1)
```

```
{
    s.o.p((char)i);
}
obj.close();
obj1.close();
obj2.close();
obj3.close();
```

<u>6)ByteArrayOutputStream:-</u>

Write a common data to a multiple file.

```
FileOutputStream obj=new FileOutputStream("as.text");
FileOutputStream obj1=new FileOutputStream("as1.text");
ByteArrayOutputStream as=new ByteArrayOutputStream();
String b[]="I am Atul";
Byte b[]=name.getBytes();
as.write(b);
as.writeTo(ob);
as.writeTo(obj);
as.flush();
as.close();
```

<u>7)ByteArrayOutputStream:-</u>

```
FileOutputStream ob=new FileOutputStream("as.text");
ByteArrayOutputStream obj=new ByteArrayOutputStream(ob);
Int i=;
while((i=obj.read())!-1)
{
    s.o.p((char)i);
}
obj.close();
ob.close();
```

<u>8)DataOutputStream:-</u>

```
FileOutputStream ob=new FileOutputStream("as.text");
DataOutputStream obj=new DataOutputStream(ob);
```

```
obj.writeInt(65);
obj.flush();
obj.close();
```

9)DataOutputStream:-

```
FileOutputStream ob=new FileOutputStream("as.text");
DataOutputStream obj=new DataOutputStream(ob);
int count=ob.available();
byte []n=new byte[count];
obj.read(n);
for(byte i:n)
{
    char l=(char)i;
    s.o.p(l);
}
```

10)FilterOutputStream:-

```
FileOutputStream ob=new FileOutputStream("as.text");
FilterOutputStream obj=new FilterOutputStream(ob);
FilterOutputStream obj1=new FilterOutputStream(obj);
String name="I am Atul";
Byte n[]=name.getBytes();
obj1.write(n);
obj1.flush();
obj1.close();
obj.close();
```

11)FilterInputStream:-

```
FileOutputStream ob=new FileOutputStream("as.text");
FilterInputStream obj=new FilterInputStream(ob);
FilterInputStream obj1=new FilterInputStream(obj);
int i=0;
while((i=obj.read())!=-1)
{
    s.o.p((char)i);
}
obj.close();
obj1.close();
```

console

get input from console . it will not display on console what you want to input .

```
imoprt java.io.*;
class atul
{
    p.s.v.m()
    {
            Console ch=System.console();
            s.o.p("Enetr user name:-");
            char cha=ch.readPassword();
            String pass=String.valueOf();
            s.o.p("Uaser name:-"+pass);
    }
}
```