MODULE 1

- Java is an object oriented programming language.
- Java is a high level language which can easily be used and understood by the user.
- Java is also called as technology.

WHAT IS JAVA TECHNOLOGY?

Java is an object oriented, platform independent (portability), multi-threaded programming environment.

HISTORY OF JAVA:

- **WHO CREATED JAVA-**

  James Gosling with his team at sun Microsystems initiated in the year 1991 with OAK. Later due to some problem the name OAK is changed to JAVA in the year 1995.

- **WHY THE NAME OAK CHANGED TO JAVA?**

  The name OAK changed to JAVA because there is already a programming language with the name OAK. So James Gosling and his team changed the name to JAVA.

- JAVA is the name of the island in Indonesia from where coffee is supplied. The icon symbol of java is hot coffee cup.

WHY JAVA IS SO POPULAR?

JAVA is so popular because we can develop the applications like device, desktop and enterprise (project).

DEVICE APPLICATION: Device applications or the programs running or handled devices such as mobile phones, printers, scanners.

DESKTOP APPLICATIONS: These applications can run only on single computers like bill generators.

ENTERPRISE APPLICATIONS: These applications can run multiple lines or multiple scanners (internet network).

Eg: -Bank transactions, railway reservations, bus reservations.

CONCEPTS OF OBJECT-ORIENTED PROGRAMMING:

1. Object
2. Class
3. Encapsulation
4. Inheritance
5. Polymorphism
6. Data Abstraction
7. Message Passing

DIFFERENCE BETWEEN C AND JAVA

- Java doesn't include the C unique statements keywords like sizeof , typedef.
- Java does not contain structures and unions like C.
- Java does not support storage classes like auto, register, extern except static.
- Java does not support signed and unsigned data types.
- Java does not support an explicit pointer type.
- Java does not support like pre-processor directives like #include, #define.
- Java supports break and continue statements but not goto statement.
- Java requires that the functions with no arguments must be declared empty parenthesis and not with void keyword as in C.

DIFFERENCE BETWEEN C++ AND JAVA

- Java does not support operator overloading, but C++ can support.
- Java does not support template classes as in C++.
- Java does not support multiple inheritance and this can be replaced using a new feature called Interface.
- Java does not support global variables and every variable and functions are declared within a class.
- Java does not support pointers.
- Java does not support destructor functions and it is replaced with a function called finalize.
- Like in C and C++ there are no header file in java and instead if that we can use packages to import the inbuilt files.
- Java supports break and continue statements but not goto statement.

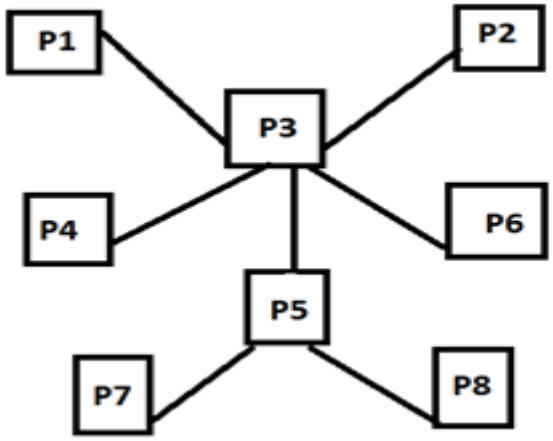
JAVA AND INTERNET

- Java is strongly associated in the internet because the fact that the first application program was written is "hot java".
- Hot java is a web browser to run applets on internet.
- Internet users can use java to create applet program and run them globally using java enabled browser such as hot java.
- Java is popularly known as internet language.
- In internet when one person want to communicate with other person for some information and if these persons having different operating systems, then they has to run in a platform independent language to run their applications and this is supported by JAVA.
- Generally, the user always requires the security about their data, therefore java can also provide security for their information.

JAVA AND WORLD WIDE WEB

- WWW(world wide web) is an open ended information retrieval system design to use in the internet's distributed environment.
- This system contains web pages. They provide both information and control.
- The web system is open ended and we navigate to a new document in any direction and this mode possible in HTML(hyper text markup language).

- The web page contains the html tags that enable us to find, retrieve, manipulate and display documents worldwide.
- Before java worldwideweb is limited to display the images and texts.
  However, the incorporation of java into web pages has made it capable of supporting of animation, graphics, games and a wide range of special effects with the support of java and therefore, the web has become more interactive and dynamic.
- Java communicates through a special tag called <APPLET>.



WEB BROWSER

- Web browser is an application that is used to retrieve the information, manipulate the information and communicate the information.
- There are different browsers like internet explorer, Mozilla firefox, google chrome, opera and so on.

JAVA FEATURES

1. Simple
2. Robust(exception handling functions)
3. Multi-threaded
4. Distributed system
5. Architecture-neutral
6. Interpretered-high performance
7. Portable
8. Secure
9. Dynamic
10. Object Oriented

- **SIMPLE:** Java was designed to be easy for professional programmer to learn and to use effectively.

- **ROBUST:** Memory management mistakes and mishandle exceptional conditions can be handled automatically by java by making memory free by garbage collector and handles the exceptions automatically by inbuilt exception handling system.

- **MULTI-THREADED:** Java supports multi-threaded programming which allows to write programs to perform multiple operations simultaneously.

- **DISTRIBUTED SYSTEM:** Java is designed for distributed environment for the internet because it handles TCP/IP protocols.
  TCP (TRANSMISSION CONTROL PROTOCOL) which is set of networking protocols used for communication on multiple computers IP (INTERNET PROTOCOL)

- **ARCHITECTURE-NEUTRAL:** The Java compiler creates an architecture neutral object file format which makes the compiled code executable on many processors with the presence of java runtime system. There is no guarantee that the programs which are running today and tomorrow in the same manner, but java achieved this problem. The main goal was WORA (write once and run anywhere) to a great extent.

- **Interpretered-High Performance:** The interpreter converts the byte code (.class file) into machine code with the help of JVM to perform well on very low power superiors.

- **Portable:** It is easy to run any java program in any system when compared to other programming language. So java can also be called as platform independent.

- **Secure:** Java secures the information without moving the data freely in and around the systems with the help of classes.

- **Dynamic:** Java programs carry the substantial amount of information during run time that is used to verify and resolve the access with the object at run time.

- **Object Oriented:** Java is purely object oriented and provides abstraction, encapsulation, inheritance and polymorphism. An object is an instance of a class and it consists of variables and related methods.

SIMPLE JAVA PROGRAM

Filename: welcome.java

```
class welcome
{
public static void main(String args[])
{
System.out.println("Welcome to my java world with smile.");
}
}
```

Compilation Command:

`javac welcome.java` //compile java file and creates byte code (`welcome.classfile`)

Execution Command:

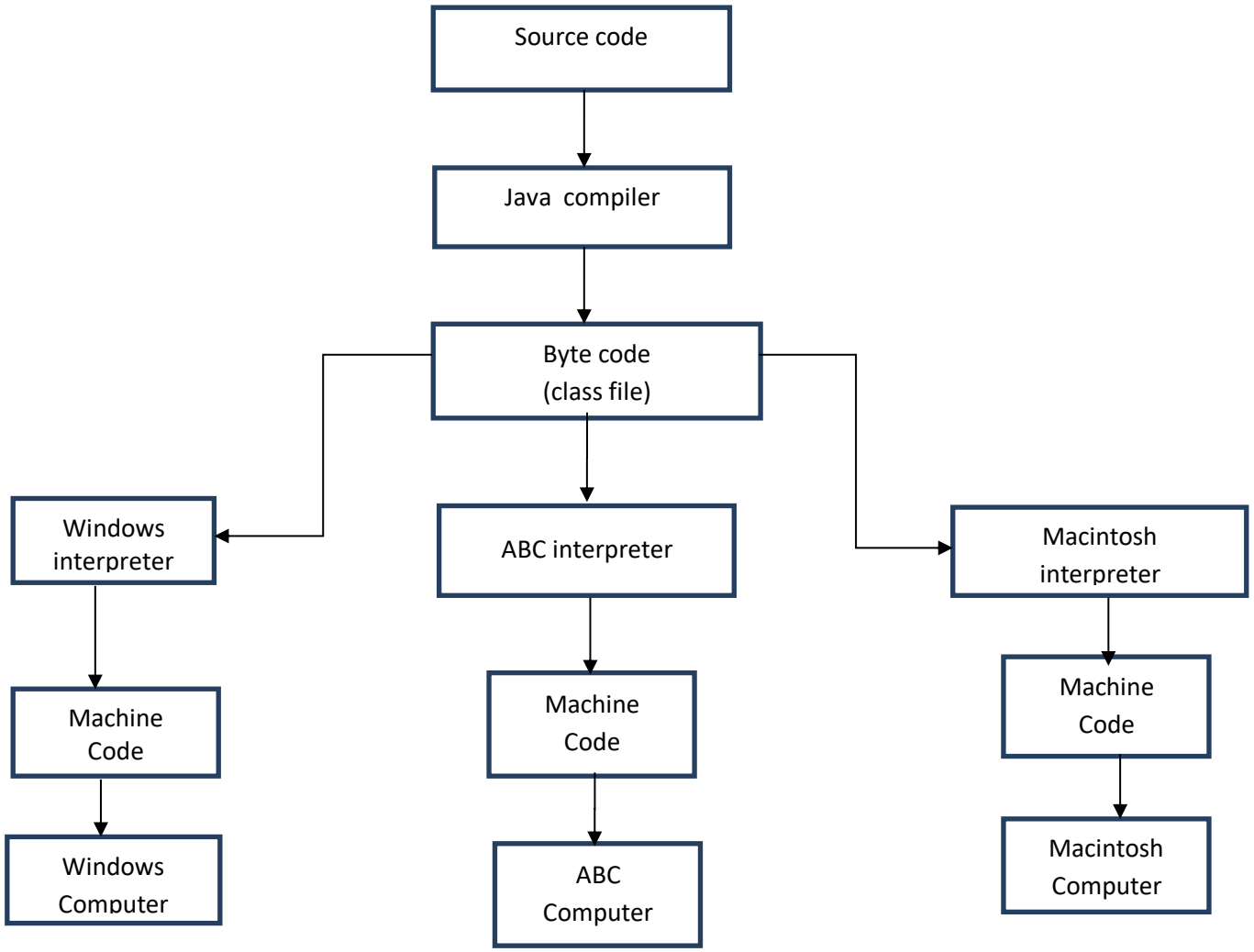
`java welcome` //interpreter translate byte code into machine code and execute the program.

Output:

```
Welcome to my java world with smile.
```

BYTE CODE (class file)

- Byte code is the output of the java compiler and the input is taken as java program or java source code.
- The output of the java compiler is not an executable code rather it is a byte code.
- Byte code is a finely optimized set of instructions designed to be executed by the java run time system, which is called as java virtual machine(JVM).
- JVM is an interpreter for byte code only.JVM needs to be implemented for each platform.
- Generally, JVM takes the byte code as input and generates the machine code with the help of java interpreter.
- We can run the JAVA in any operation system because of the interpreter present in every operating system therefore, the JVM generates the machine code (byte code) with the help of interpreters.



JAVA ENVIRONMENT: -

- Java environment includes large number of development tools, classes and methods.
- Development tools are the part of the system like JDK, JRE, JVM, JSL (Java Standard Library), classes, methods and so on.
- The classes and methods are the part of java standard library (JSL) and it is also known as application program interface (API).

JAVA DEVELOPMENT KIT (JDK):

The JDK comes with a collection of tools that are used for developing and running java programs. Some standard JDK tools are:-

1. appletviewer
2. javac
3. java
4. javap
5. javadoc
6. javah
7. jdb
8.rmic

1. appletviewer:
It enables us to run java applets which are used to run internet supported program.

2. javac:

It is a java compiler which translates the java source code into byte code that the interpreter can understand.

3. java:

It is a java interpreter which runs the applets and applications by reading and interpreting byte code files into machine code.

4. javap:

It is a java disassembler which enables us to convert byte code files into program description (Abstract of the source code).

5. javadoc:

It creates html format documentation from java source code files.

6. javah:

It produces the C or C++ header files to use with native methods that mean javah acts like a interface between java and C/C++.

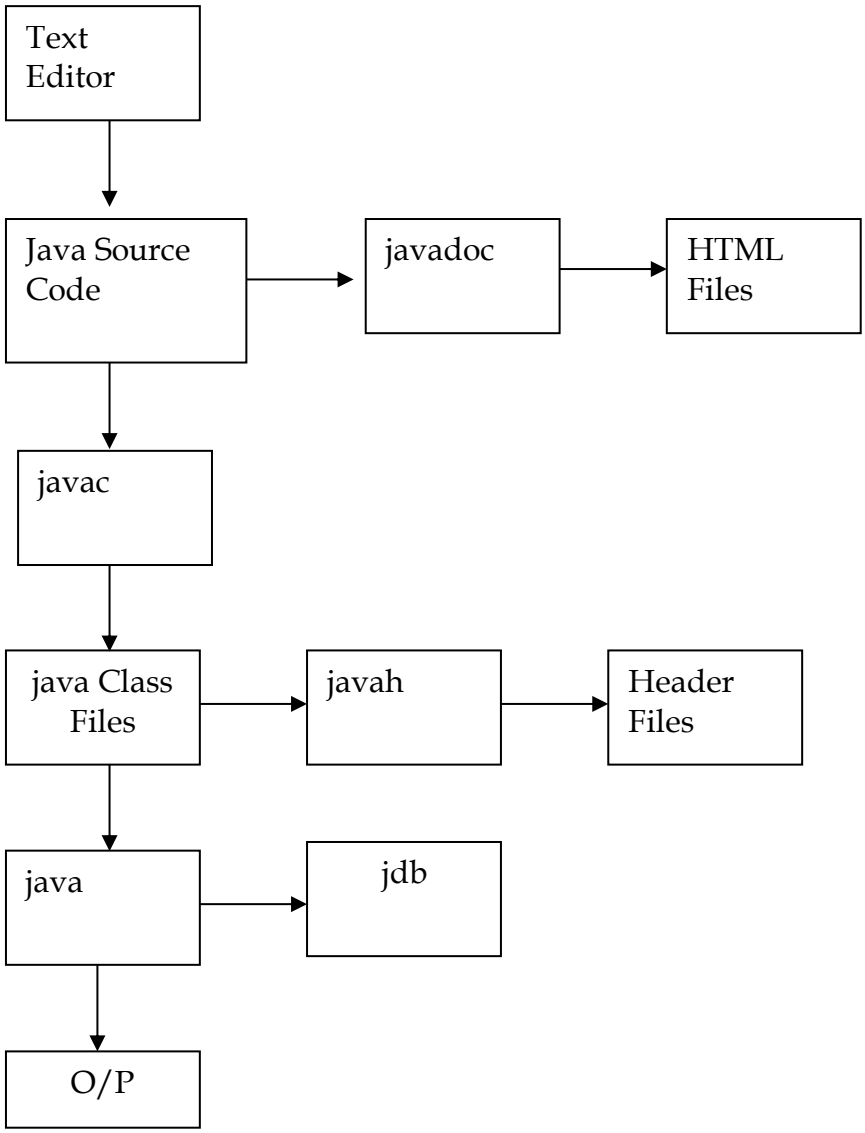
7. jdb:

It is a java debugger which helps to find the errors in the program.

8.Rmic:

Create skeletons and stubs for remote method invocation.

Procedure (or) steps to execute a Java Program



Java Standard Libraries (JSL) (or) Application Programming Interface.

- There are several classes and methods grouped into number of packages and some of the standard packages are-

1. java.lang.*
2. java.io.*
3. java.net.*
4. java.util.*
5. java.applet.*
6. java.awt.*

Language support Package(java.lang.*)

- It is a collection of classes and methods which are required for implementing basic java features.
  Ex. Mathematical functions like square root, power and so on….

Input Output Package (java. io.*)
This is used for input and output manipulations like reading and printing the values.

Applet Package (java. applet.*)
This includes a set of classes that allow us to create java applets like Internet programs.

Networking Package (java.net.*)
A Collection of classes for communicating with other computers via internet (LAN or WAN)

AWT package(java. awt.*)
The abstract window tool kit package contains classes that implement platform independent graphical user interfaces like animation.

Utility Package (java. util.*)
A Collection of Classes to provide utility functions such as date and time functions.

JRE (Java Runtime environment)
- It is set of software tool for development of java application.
- It provides the minimum requirements for executing java application.
- It primarily comprises the following.
- JVM (Java virtual machine)
- Runtime class libraries
- Uses interface tool kits
- Deployments technologies.

JVM
It is a program that interprets the intermediate java byte code and generates that desired output. It is because of byte code and JVM concepts the programs written in Java are highly portable.

Runtime Class Libraries (Package)
There are a set of core class libraries that are required to execute Java program.

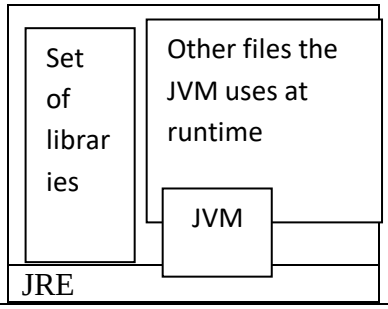
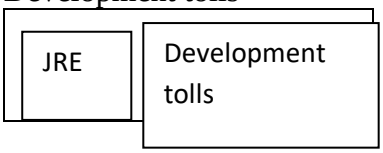
User Interface tool kits
AWT and SWING are examples of tool kits that support various input methods for the user to interact to the application program like bill generation calculators, and so on.

Deployment Technologies
JRE comprises the following deployment technologies

1. Javaplug – in :-
It enables the execution of Java applet of browses.

2. Java Web Start:-
It enables remote deployment of an application with web start users can launch an application directly from the web browser without going through installation procedure.

Differences between JWM, JDK and JRE

JVM	JRE	JDK
It executes the Java byte code like loads code, verifies code, executives code providesruntime environment	It is implementation of JVM i.e. JRE = (Set of libraries) + other files the JVM uses at runtime) 	JDK = JRE + Development tolls 

Structure of a Java Program

Documentation section	Suggested
Package Statement	
Import Statement	Optional
Interface Statement	
Class definition	

Main method Class
{
}
	Essential

- Documentation -> /* about Java program */
- Package ->package student ;
- Import Stunt ->import java.lang.*;
- Interface definition ->interface student
  {
  Statement(s);
  }
- Class definition -> class student
  {
  Statement(s);
  }

- Main Method Class

```
->class student
{
public static void main(String args[])
{
```

```
                    Statement1;
                    Statement2;
        }
                }
```

TOKENS

There a five types of tokens –

1. Keywords
2. Identifiers (Variable name, function name)
3. Constants /Literals
4. Operators
5. Separators

1. Keywords

Keywords are reserved words, which are used to perform a specific task and we have 53 keywords in Java.

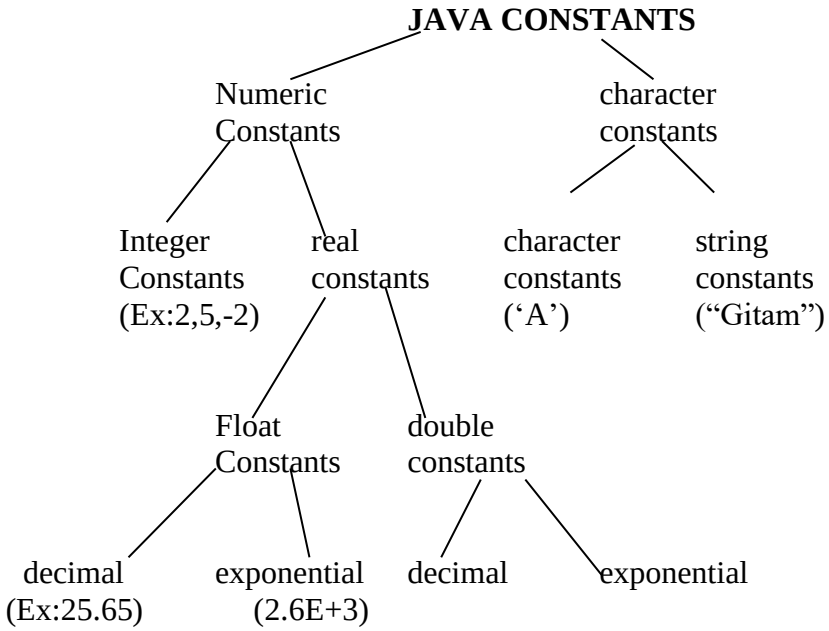
abstract	assert (added in J2SE.1.4)	boolean	break
byte	case	catch	char
class	const(un-used)	continue	default
do	double	else	enum
extends	final	finally	float
for	goto(un-used)	if	implements
import	instanceof	int	interface
long	native	new	package
private	protected	public	return
short	static	strictfp(addedin J2SE.1.2)	super
switch	synchronized	this	throw
throws	transient	try	void
volatile	while	true	null
false			

NOTE:- **1.** True, false, null are reserved for literal values.
2. Goto and const are reserved in java but have no function.

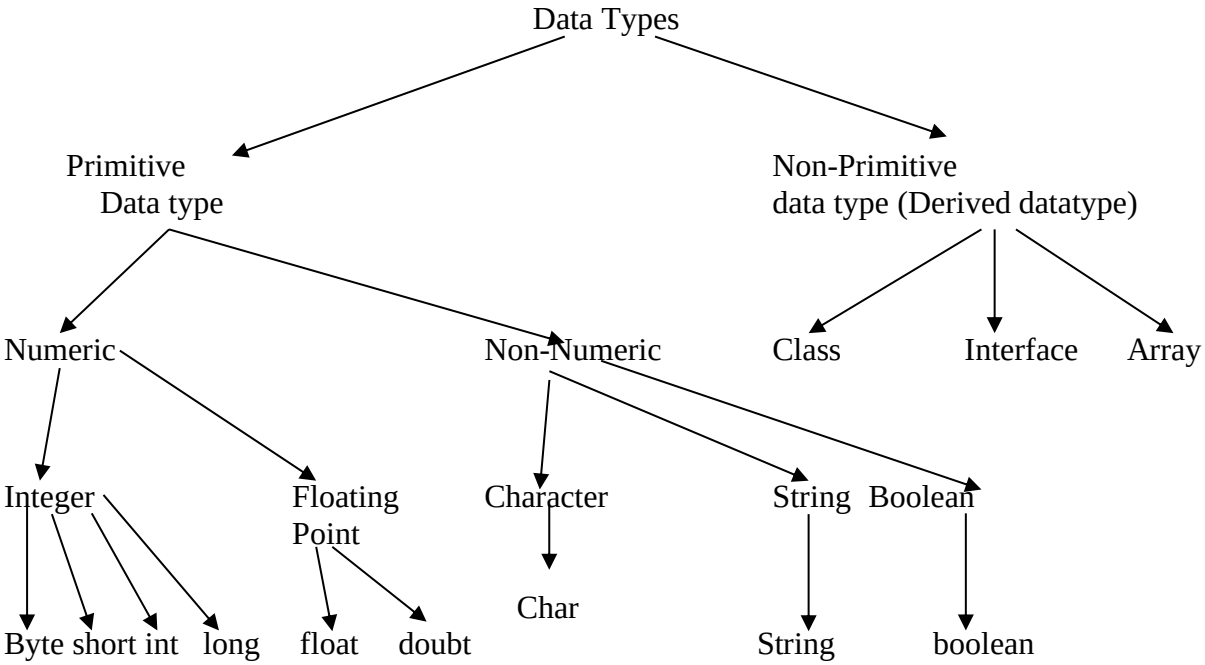
Identifiers:

- Identifiers are the variables names, function names, class names.
- Variables are used to retrieve the data and to perform some operations.
- While declaring a variable the first letter of the variable name should start with an alphabet or underscore.
- No special symbols are allowed to declare as a variable name except underscore.
- No spaces are allowed in between the variable names

Ex:-	**valid**	**invalid**
	ab	2a
	a_b	a@b
	a2	ab
	_ab	


JAVA CONSTANTS
Numeric Constants
character constants
Integer Constants (Ex:2,5,-2)
real constants
character constants (‘A’)
string constants (“Gitam”)
Float Constants
double constants
decimal (Ex:25.65)
exponential (2.6E+3)
decimal
exponential

DATA TYPES


Data Types
Primitive Data type
Non-Primitive data type (Derived datatype)
Numeric
Non-Numeric
Class
Interface
Array
Integer
Floating Point
Character
String
Boolean
Char
Byte
short
int
long
float
doubt
String
boolean

Memory Size and min-max values

Date type	Memory Size (Byte)	Minimum Value	Maximum Value
Byte	1	-128	+127
Short	2	-32768	+32767
Int	4	-2147483648	+2147483647
Long	8	-9223372036854775808	+9223372036854775807
Float	4	3.4E-38	3.4E+38
Double	8	1.7E-308	1.7E+308
Char	2	0	65535

Steps to create, compile and execute the java program:

Step-1: Open command prompt (dos) to run command i.e., window key + R and type cmd and press enter.
Step-2: Notepad filename .java to create an empty java file or open the existing file.
Step-3: Type the given java program.
Step-4: Save the program and minimize the window.
Step-5: Compile the java program by using command javac filename.java
Step-6: If there is no error execute the java program by executing the program by using the command java classname.

Difference between print and println:

- The print function prints the string or message in the current position or in the current line. println function also prints the message or string in the current line, and then the pointer position to the next line.
- println function is same as print function with the new line command at the ending of the string.

Ex:- `System.out.println("Hi how r u");` **both are same**
`System.out.print("Hi how r u \n");`

ARRAYS

- An array is a group of variables that referred by a common name.
- An array is a homogeneous data type which contains multiple no. of variables with same name and with same data type with continuous memory allocation.
- Syntax :-

```
Datatype array_name[] = new datatype[size];
```

Method for declaring an array

Method1:

```
int a[]=new int[5];
```

Method 2:

```
 int a[];
a=new int[5];
```

INITIALISING OF ARRAYS

Syntax –

```
datatype array[]={val1, val2, val3, …};
```

Ex:-

Method 1 :–

```
int a[]={10,20,30,40};
```

Method 2 :–

```
int a[]={10,20,30,40};

     int b[];

     b=a;
```

Method 3: –

```
int a[] = new int[4];
     a[0]=10;
     a[1]=20;
     a[2]=30;
     a[3]=40;
```

Method 4 :–

```
 int a[]=new int[4];
for(int i=0;i<n;i++)
Input_statement(Ex: Scanner, BufferedReader etc.,)
```

Program:

```
class array
{
     public static void main(String args[ ])
     {
        int a[ ]={10,20,30,40};
        int b[ ];
        b=a;
        int c[ ]=new int[4];
     c[0]=50;
     c[1]=60;
     c[2]=70;
     c[3]=80;
     System.out.println("given list of elements");
for(int i=0;i<4;i++)
System.out.println("a["+i+"]="+a[i]);
     System.out.println("given list two elements");
          for(i=0;i<4;i++)
System.out.println("b["+i+"]="+b[i]);
     System.out.println("given list three elements");
     for(i=0;i<4;i++)
          System.out.println("c["+i+"]="+c[i]);
     }
}
```

PROGRAM

```
class sort
{
public static void main(String args[ ])
{
int a[]={20,10,40,30,80};
System.out.println("Elements before sorting are :");
for(int i=0;i<5;i++)
     System.out.print(a[i]+"  ");
for(i=0;i<5;i++)
{
     for (int j=1;j<5-i;j++)
     {
          a[j-1]= a[j-1]+a[j];
          a[j]=a[j-1] - a[j];
          a[j-1]= a[j-1]- a[j];
     }
}
System.out.println(" After sorting the list is: ");
for(i=0;i<5;i++)
     System.out.println(a[i]+"  ");
}
}
```

ARRAY LENGTH

We can find the array length by using the arrayname . length.

```
int len = arrayname.length;
```

Program

```
class length_array
{
public static void main(String args[])
{
     int a[ ] = {50,10,40,20,5};
     int len = a.length;
     System.out.println("Length of given array list is");
}
}
```

- WAP to perform linear search by using the given list of elements and display the key element with index position and if the element is not found display the element not found.

Linear Search Source code :

```
import java.util.*;
class search
{
public static void main(String args[ ])
   {
  Scanner S = new Scanner(System.in);
int a[ ]={10,20,30,40,50};
  int key,flag=0,pos;
  System.out.println ("The given list of elements are: ")
  for (int i=0;i<5;i++)
  System.out.println(a[i]+" ");
  System.out.println("Enter the element to search");
  key = S.nextInt();
  for(i=0;i<5;i++)
  {
       if(key == a[i])
       {
            flag=flag+1;
            pos = i+1;
            break;
       }
  }
  if(flag == 0)
       System.out.println(key + "is not found");
  else
       System.out.println(key+ "is found in pos" + pos);
  }
}
```

2D ARRAYS

Declaration Syntax:

```
datatype arrayname[ ][ ]=new datatype[S1][S2];
```
Where S1 and S2 are array Size(S1=row, S2=column).

```
Ex:- int a[][]= new int[10][10];
     float f[][]= new float[3][3];
     char ch[][]= new char[2][5];
```

- 2D array is also homogeneous data type with same variable name with continuous memory location.
- 2D array is also called as multidimensional array and this contains the data in the form of rows and columns.

Initialization

```
datatype arrayname[][] = {{Val1,Val2,Val3},{Val4,Val5}};
```

Ex:-

Method 1:

```
int a[][ ] = {{10,20,30},{40,50,60}};
```

(or)

Method 2:

```
int b[][];
b = a;
```

(or)

Method 3:

```
int a[][] = new int[2][2];
a[0][0] = 10;
     a[0][1] = 20;
     a[1][0] = 30;
     a[1][1] = 40;
```

(or)

Method 4:

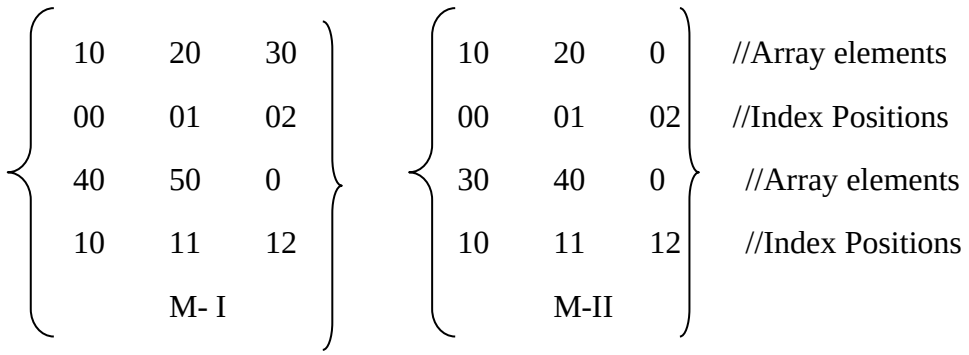
```
int a[ ][ ] = new int[2][2];
     for(int i=0;i<2;i++) //Looping statements
          for(int j=0;j<2;j++)
              a[i][j]=S.nextInt(); //Input statements
```

Invalid Initialization of 2D array

```
int a[2][3] = {10,20,30,40,50};//M-I
```

This method is not preferred to initialize in 2D arrays therefore, we have to specify the elements of each row with flower brackets as shown below.

```
int a[2][2] = {{10,20},{30,40}};//M-II
```



PROGRAM PRINTING OF 2D ARRAY

```
class  2Darray
{
public static void main(String args[ ])
{
     int a[][]={{10,20,30},{40,50,60}};
     System.out.println("Given2D array elements are");
     for (int i=0; i<2; i++)
     {
          for (int j=0;j<3;j++)
               System.out.print(a[i][j]+" ");
          System.out.println( );
     }}}
```

Variable size arrays

In java it is possible to declare different column size in each row.
Syntax –

```
datatype array[ ][ ]=new datatype[size][];
array[0] = new datatype[int_size];
array[1] = new datatype[int_size];
array[2] = new datatype[int_size];
```

Program

```
import java.util.*;
class test
{
     public static void main (String args[ ])
     {
          int i,j;
          Scanner S= new Scanner (System.in);
          int a[][] = new int[3][];
          a[0] = new int [2];
          a[1] = new int [4];
          a[2] = new int [3];
          for (i=0; i<3; i++)
          {
               System.out.println("Enter"+a[i].length+"in"+
                    (i+1)+"row") ;
               for (j=0; j<a[i].length ;j++)
```

```
                    a[i][j] = S.nextInt( );
                System.out.println("given elements are:")
                for (i=0;i<a.length; i++)
                {
                    for (j=0; j<a[i].length; j++)
                        System.out.println(a[i][j]+ "  ");
                    System.out.println(  );
                }
            }
        }
}
```

Note :

- In the above program `a.length` returns no. of rows in a 2D array.
- And `a[i].length` returns the length of each row i.e., no. of columns in each rows.

For each loop

- For each loop is also called as enhanced for loop.
- For each loop is the extended language feature introduced with J2SE 5.0 version.
- This feature helps to retrieve the array of elements efficiently by using array index.
- We can also use this feature to eliminate the attraction in for loop and to retrieve the elements from the list.

Syntax

```
 for(datatype var:array_name)//Assigns the value of each element of array
{
    Statement1;
    Statement2;
}
```

Advantages: It makes the code more readable. It eliminates the possibility of programming errors.

Program

```
class foreachloop
{
    public static void main (String args[ ])
    {
        int a[] = {10,20,30,40};
        int i;
        System.out.println("using normal for loop");
        for(i=0;i<4;i++)// using for loop
            System.out.println (a[i] + " ");
        System.out.println ("using for each loop");
        for(int j:a)//using enhanced for loop
        {
            System.out.println(j+" ");
        }
```

```
    }
}
```

Difference between for loop and for each loop

For loop	For each loop
1. It contains 3 components and separated by a semicolon(;)	1) It contains only 2 components and separated by a colon(:)
2. We use initialization, condition & updation as 3 components	2. We use variable declaration and existing arrayname as 2 components.
3. We can do any operations by using for loop, like with arrays or without arrays.	3. We can access only array elements and we cannot perform the operation other than arrays.
4. Manually, we have to increment or decrement the index position of an array.	4. System automatically increments the index position of an array.
5. Condition must specify to terminate the loop	5. The loop terminates automatically whenever it reaches to the last element of the array.

Accessing 2D dimensional array's elements using or each loop (enhanced for loop)

Syntax:-

```
for (datatype array[ ]:array)
{
     for (datatype var : array1)
     {
          Statement1;
          Statement2;
     }
}
```

Example-

```
class twoDArray
{
     public static void main(String args[ ])
     {
          int a[][]={{10,20,30},{40,50,60}};
          for(int i[]:a)
          {
               for(int j:i)
                    System.out.print(j);
               System.out.println();
          }
     }
}
     O/p:- 10 20 30
           40 50 60
```

Note: To access 2D array to enhance the for loop we have to used the nested for each loop concept.

The outer loop is used to store the 2D array element into single dimensional array row by row.
Similarly, the inner loop is used to store the 1D array elements into the variable one by one.

Access array of strings using enhanced for loop

```
class String_Array
{
   public static void main (String args[])
   {
     String names[]={"Ram","Sita","Maruthi","Laxman"};
    for(String s: names)
        System.out.println(s);
   }
}
```

COMMAND LINE ARGUMENTS

- Command line arguments are passing the values to the main program at the time of program execution.
- In java the main function contains only one argument i.e. array of strings.
- The reason behind why only strings why not other datatype is because it is easy to convert the string values into various primitive datatypes by using parsing technique

Syntax :

```
public static void main (String args[ ])//command line arguments
{
     Statement1;
     Statement2;
}
```

Program

```
class Commandline
{
pubic static void main (String args[ ])
{
System.out.println("List of names through commandline are:");
for(String s: args)
     System.out.println(s);
}
}
```

Input:-

```
C:\>java filename Sachin Virat Dhoni Dravid
```

Output:-

```
List of names through commandline are
     Sachin
```

```
Virat
Dhoni
Dravid
```

Passing methods to convert strings into primitive values

```
1. int  i= Integer.parseInt(str);
2. float f = Float.parseFloat(str);
3. double d= Double.parseDouble(str);
4. long l=Long.parseLong(str);
5. short s=Short.parseShort(str);
6. byte b= Byte.parseByte(str);
7. boolean bl=Boolean.parseBoolean(str);
8. char ch=stringvariable.charAt(0);//non-parsing technique.
```

- The parsing methods are used to convert the string values into primitive values.
- There is no parsing technique to convert string into character, therefore we have to use charAt( ) method to take a character value for a string from the given position

Basic I/O streams

1. Scanner
2. BufferedReader
3. InputStreamReader

Scanner

- Scanner is a predefined class in the package "java.util" which is used to read the input values through keyboard by passing the input stream System.in as a parameter in the constructor of scanner.
- In system.in is an input stream which is typically connected to keyboard, input of console program.

  Syntax:

  ```
  Scanner S= new Scanner(System.in);
  ```

  Scanner can read the input values from the keyboard or from the text file.

  Example:

  ```
  Scanner S= new Scanner("test file.text");
  ```

- The Scanner class can read and parse any type of primitive values like int, float, char, string and so on.
- The Scanner class occupies 1 kilo byte of memory for the input values.
- Before scanner there are some other concepts available to read the values from the keyboard, like InputStreamReader and BufferedReader.
- The Scanner is introduced in JDK 1.5 version

List of Methods in Class Scanner

```
int n= S.nextInt( ); // Reads integer values
long l=S.nextLong( ); // Reads dong values
```

```
short s= S.nextShort( ); // Reads short values
byte b = S.nextByte( );// Reads Byte values
boolean bl = S.nextBoolean( ); // Reads Boolean values
float f = S.nextFloat( ); // Read float values
double d = S.nextDouble( ); // Reads double values
String str1 = S.next( ); // Reads string without spaces
                         (ex: Ramlaxman)
String str2 =S.nextLine( ); // Reads string with spaces
                         (ex: "Ram laxman")
char ch=S.next().charAt(0); // Reads single character
```

WAP to read and disply student details like roll no, name, branch, gender and percentage.

Source code

```
class student
{
public static void main(String args[])
{
     String name, branch;
     int rno;
float per;
    char gndr;
Scanner S = new Scanner(System.in);
System.out.println("enter name");
name = S.nextLine( );
    System.out.println("enter branch");
branch=S.next();
System.out.println("enter rno");
rno=S.nextInt();
System.out.println( " enter per);
per = S.nextInt( );
System.out.println ("enter ur gender /m/f)");
gndr= S.next( ).charAt(0);
System.out.println( "the given student details are ");
System.out.println("name is " + name);
System.out.println("branch is " + branch);
System.out.println("roll no is" + rno);
System.out.println("percentage is " + per);
System.out.println("gender is " + gndr);
}
}
```

- WAP to read and display employee details like empno, emp name, emp gender, emp salary, designation and total experience

  Source Code

```
class employee
{
public static void main (String args[ ])
{
     String name,designation;
     char gndr;
     int id, salary, expr;
     Scanner S=new Scanner(System.in);
     System.out.println("enter name");
     name = S.nextLine();
     System.out.println("Enter designation");
     designation = S.nextLine();
     System.out.println("Enter employee id");
     id = S.nextInt( );
     System.out.println("Enter salary");
     salary = S.nextInt( );
     System.out.println("Enter total experience");
     expr = S.nextInt( );
     System.out.println("enter gender");
     gndr = S.next().charAt(0);
     System.out.println("given details of the employee is :");
     System.out.println("Name" + name);
     System.out.println("Designation" +designation);
     System.out.println("Employee " + id);
     System.out.println("Salary " + salary);
     System.out.println("total experience" + expr);
     System.out.println("gender" + gndr);
}
}
```

Reading a text file using scanner

- In scanf we can read the values from the keyboard as well as text file.
- If the input stream is system.in then we can read the values from the keyboard.
- If the input stream is file object, then we can read the content or data from the file.

How to create object to a file?

Syntax –

```
File fobj = new File("source path of text file");
```

Ex:- `File fobj = new File("C:\\ Javap\\ abc.txt");`

- File is a predefined class which is defined in the package `java.io.*;`
- While executing the program on files there is a chance to occur exceptions like file not found exception therefore, manually we have to throw an exception without handling.

Program

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

```
class scannertext
{
Public static void main(String args[])throws IOException
{
     File ft = new File("C:\\abc.txt");
     Scanner S=new Scanner(ft);
     int linenumber=1;
     while(S.hasNextLine( ))//Reads upto end of the file.
     {
          String line=S.nextLine( ); //Reads each line
          System.out.println("Line"+linenumber+":line"+
                                                  linenumber++);
     }
}}
```

Input Stream Reader

- Input stream reader is a pre-defined class which is defined in the package **`java.io.*;`**
- This class reads the input values in the form of bytes, and converts to character like a bridge.
- This class reads bytes and decide them into characters using a specified char set.

Methods to read the value

1. `int read()` - This method reads a character and returns integer value (ASCII value of a character).

Ex:- If the input is A, them it returns the output as 65.

Exception Occurred:

While running this program there's a chance to occur some runtime exception like input not available or memory is not available, therefore we have to throw manually an IOException (Input output exception).

Program:-

```
import java.io.*;
class input
{
     public static void main(String args [ ])throws IOException
     {
     InputStreamReader imp = new InputStreamReader(System.in);
     System.out.println("Enter a character");
     int n=imp.read( ); // returns ASCII value of input character
     char ch=(char)n;//Explicit type casting to convert int to character
     System.out.println("given character is"+ch);
     System.out.println("Its ASCII value is"+n);
     }
}
```

- WAP to read 2 single digit integer no.'s and perform various arithematic operation. (using InputStreamReader)

- WAP to read more than one digit of integer no.'s and perform various arithematic operations using input stream reader.

Buffered Reader

- Buffered reader is a pre-defined class which is defined in the package `java.io.*` to read the input values from the keyboard in the form of characters and strings.
- Buffered reader is the early version in java to read the input values.
- Buffered reader have a capacity of reading the large memory size of input values like 8 kilobytes .This is because in internet, most of the cases the values in memory size will be big.
- Buffered reader can only read the values with the help of input stream reader class. To read from the keyboard with the support of input stream called System .in.

Method-1

```
InputStreamReader in = new InputStreamReader(System.in);
 BufferedReader br=new BufferedReader(in); ------ (1)
```

Method:2

```
BufferedReader br=new BufferedReader(new InputStreamReader
(System.in));----------(2)
```

- The major drawback of BufferedReader is we cannot read all the primitive values directly. Therefore, manually using different parsing methods we have to convert the string values into primitive values.
- Some of the parsing methods to convert string into primitive values are:

```
int i = Integer.parseInt(str);
float f = Float.parseFloat(str);
double  d = Double.parseDouble(str);
long l = Long.parseLong(str);
short s = Short.parseShort(str);
byte b = Byte.parseByte (str);
boolean bl = Boolean.parseBoolean(str);
char ch = StrngVrable.charAt(0);
```

Input Methods Available in Buffered Reader

1. read () – read function is used to read a single character and returns ASCII value of the character.
2. readLine () - which is used to read a string.

Eg:-
```
int i = br.read();
char ch = (char)br.read( );
String s = br.readLine( );
```

Program

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

```
class bufferedreader
{
public static void main (String args[])
{
String s;
InputStreamReader in = new InputStreamReader(System.in);
BufferedReader br = new BufferedReader(in);
System.out.println("enter ur name");
String name = br.readLine( );
System.out.println("enter ur gender");
char gndr = (char)br.read( );
br.readLine( );
System.out.println("enter ur age");
S=br.readLine( );
Short age=Short.parseShort(s);
System.out.println("enter roll no");
S=br.readLine( );
int rno=Integer.parseInt(s);
System.out.println("enter ur percentage");
float per = Float.parseFloat(br.readLine());
System.out.println("enter cell no");
long cell = Long.parseLong(br.readLine());
System.out.println("given student details are :");
System.out.println("name is "+ name);
System.out.println("roll is "+rno+"/n age is "+age);
System.out.println("gender is"+gndr+"\npercentage is"+per);
System.out.println("cell no is" + cell);
}
}
```

- WAP to read & Display patient details like patient name, bed no, age, gender, disease, doctor name and total bill, using buffered reader metod-2

Source code

```
import java.io.*;
class patientdetails
{
public static void main String args[])throws IOException
{

String s;
BufferedReader br=new BufferedReader(new
                        InputStreamReader(System.in));
System.out.println ("enter ur name");
  String name = br.readLine();
System.out.println ("enter bed no.");
  short bed = Short.parseShort(br.readLine());
System.out.println (" enter ur age");
short age=Short.parseShort(br.readLine());
System.out.println("enter ur gender");
char gen =(char) br.read();
br.readLine();
```

```
System.out.println (enter disease name”);
  String dis = br.readLine();
System.out.println (“enter doctor’s name”);
  String doc = br.readLine( );
System.out.println (“enter total bill”);
long bill = Long.parseLong(br.readLine());
System.out.println(given patient details are;”)
System.out.println(“bed no is :” + bed);
System.out.println(“age is : ”+age);
System.out.println(gender is :” + gen);
System.out.println(“disease name is : ”+dis);
System.out.println(“disease name is : ”+doc);
System.out.println(“total bill : ” + bill);
}
}
```

BUFFERED READER Vs SCANNER:

1. BufferedReader only reads the data, but Scanner reads the data as well as parse the data.
2. We can only read characters and strings using buffered reader whereas we can read characters, strings and other primitive values in Scanner.
3. Buffered reader is older than Scanner, where buffered reader exists from JDK1.1 version, while scanner is added in JDK 1.5 version.
4. The buffer size of buffered reader is 8 kilobytes as compared to the scanner which is 1 kilobyte.
5. Buffered reader is more suitable for reading the file with long strings while scanner is more suitable for reading small strings using input from command prompt.
6. Buffered reader is synchronised but scanner is not synchronized, which means we cannot shape scanner among multiple threads.
7. Buffered reader is faster than scanner because it does not spend time on parsing.
8. BufferedReader is a pre-defined class defined in the package “java.io.*” whereas Scanner is a pre-defined class defined in the package “java.util.*”

STRING CLASS METHODS

- `String s3=s1.concat(s2);`//concatenate the string S1 & S2 & store in S3.
- `int len = s.length();` // returns length of the string.
- `char ch = s.charAt(0);` // returns 0th character from string
- `int cmp=s1.compareTo(s2);`//returns 0 if same else non-zero (case sensitive)
- `int cmp=s1.compareToIgnoreCase(s2);` // returns 0 if same else non-zero (by ignoring case sensitive)
- `boolean bl=s1.equals(s2);`//returns true if equal else false (case sensitive)
- `boolean bl=s1.equals(s2);` // returns true if equal else false (case sensitive)

- `boolean bl=s1.endsWith(s2);`// returns true if main strings end with substrings
- `boolean bl=s1.endsWith(s2);`//returns true if main strings ends with substring s2
- `int n=s1.indexOf(s2);` //returns the starting index pos of s2 in main string s1
- `int n=s1.lastIndexOf(s2);` // returns the last index pos of s2 in main string s1.
- `String s1=s.replace(char1,char2);` //replaces the old character char1 new character char2 and returns to s1.
- `String s4= s.toUppercase();`//converts lower case to upper case.
- `String s2=s.toLowercase();` // converts to upper case to lower case.
- `String substr=s.substring(start_indexpos)`//creates substring from given index position to last.
- `Stringsubstr=s.substring(start_indexpos, end_indexpos);`//creates substring from given start position to end position.
- `S.getchar(startpos,endpos,char_arrayname,strtindexpos_Arrayname);`//convert string to char array.
- `String Arrayname = str.split(delimiter);` where delimiter is any character like `'.' , ',' , ' : '`...//splits the string into multiple strings.
- `Str.trim;` //remove the spaces before & after the string but not in middle.

LAB PROGRAMS:-

Given are 2 one dimensional arrays A ,B which are sorted in ascending order WAP to merge them into a single sorted array, (that contains every item from array's A &B in ascending order

SOURCE CODE

```
import java.io.*;
class mergesort
{
public static void main (String args [])throws IOException
{
     int l1,l2,l3,i,j,k;
     int A[],B[],C[];
     BufferedReader br=new BufferedReader (new
                         InputStreamReader(System.in));
     System.out.println("enter range of A&B");
     l1 = Integer.parseInt(br.readLine());
     l2= Integer.parseInt(br.readLine());
     A[] = new int[l1];
     B[] = new int[l2];
     C[] = new int[l1+l2];
     System.out.println("enter array a in sorted order");
     for (j=0; j<l2;j++)
          A[i]=Integer.parseInt(br.readLine());
     System.out.println("enter array B in sorted order:");
     for(j=0;j<l2;j++)
          B[j]=Integer.parseInt(br.readLine());
     i=j=k=0;
```

```
        while(i<l1&&j<l2)
        {
              if (A[i]<B[j])
              {
                    C[k] = A[i];
                    i++;
                    k++;
              }
              else if (A[i]>B[j])
              {
                    C[k]= B[j];
                    j++;
                    k++;
              }
              else
              {
                    C[k]=A[i];
                    i++;
                    k++;
                    C[k]=B[j];
                    j++;
                    k++;
              }
        }
        while(i<l1)
        {
              C[k] = A[i];
              i++;
              k++;
        }
        while(j<l2)
        {
              C[k] = B[j];
              j++;
              k++;
        }
        System.out.println("After merging array A& B the result is: ");
        for(i=0; i<k; i++)
              System.out.println(C[i]);
        }
}
```

DIFFERENCE BETWEEN A STRING WITH NEW OPERATOR AND WITHOUT NEW OPERATOR

Method 1:

```
String s1 = "abcd";
```

Method2:

```
String s2 = new String("abcd");
```

- In Method1, the JVM first checks the content whether it is available in string constant pool or not. If it is there, then the JVM creates the reference object to the existing object.
- If it is not there in the pool, then creates a new object.

- In Method2, by using new operator irrespective of the content it always equates a new string object.

Ex1:-
```
String S1 = "abcd";
String S2 = "abcd";
```
Ex2:-
```
String S1 = new String ("abcd");
String S2 = new String ("abcd");
```

Therefore S1= =S2 gives false result

Note:- 1) "==" checks the object address but not value

2) Irrespective of the object address to compare the values of two strings it is preferred to use the string, methods compare to or "compare to ignore" use.

STRING BUFFER

- In java, string buffer class is used to create mutable string (modifiable).
- The string buffer class in java is same as string class but it is mutable i.e, it can be changed.
- In java, string buffer class is thread safe i.e multiple threads cannot access it simultaneously, that means string buffer is synchronized, therefore it is safe and will results in order.

  `StringBuffer sb = "IT";` // Invalid

  `StringBuffer sb=new StringBuffer("IT");` // Valid

  CONSTRUCTION DESCRIPTION:

  1. `StringBuffer sb = new StringBuffer();` // creates an empty string buffer with the initial capacity of 16
  2. `StringBuffer sb = new StringBuffer(str);` // creates string buffer with the specified string
  3. `StringBuffer sb = new StringBuffer(length);` //creates an empty string buffer with specified capacity or length.

METHODS IN STRING BUFFER CLASS

1. `str1.append(str2);` // appends str2 into str1.
2. `str1.insert(index_pos,substring);` //inserts substring in the index pos of str1.
3. `str1.replace(strt_index,end_index,substring);`//replaces str1 from strt_index to end_index with substring)
4. `str1.delete(strt_index,end_index);`//delete the main string from strt_index to end_index.
5. `str.reverse();`//reverse the main string str.
6. `int i=str.capacity();` //returns the current capacity (range)
7. `str.ensureCapacity(int_min_capacity);`//used to increase with minimum capacity

STRING BUILDER

- String builder is a class in java used to create mutable string(modifiable) the java string builder is same as string buffered class except that which is non-synchronized and it is available since JDK 1.5

SYNTAX:-

```
StringBuilder sb= "IT";
StringBuilder sb = new StringBuilder("IT");
```

CONSTRUCTOR

1. `StringBuilder sb = new StringBuilder();`//creates an empty string builder with initial capacity of 16.
2. `StringBuilder sb = new StringBuilder(str);`// creates the builder with the specified string.
3. `StringBuilder sb = new StringBuilder(length);`// creates an empty string builder with specified capacity as length.

NOTE:- The methods which are used in StringBuffer are also applicable in StringBuilder.

STRING	STRING BUFFER
1. If the content is fixed and won't change frequently then we can use String.	1. If the content is not fixed and keep on changing but thread safety is required (synchronized thread) then we can use String Buffer.
2. We can read the strings from key board using different input methods directly.	2 . We cannot read the strings directly from the keyboard using different input methods therefore, we can use other alternative methods to read the Strings.

- WAP to read and display string using string buffer

SOURCE CODE:-

```
import java.util.*;
class stringbuffer
{
     public static void main(String args[])
     {
          StringBuffer sbr = new StringBuffer();
```

```
        Scanner S=new Scanner(System.in);
        System.out.println("enter a string");
        sbr.append(S.nextLine());//reads string and appends to sbr
        System.out.println("Given string is "+sbr);
    }
}
```

- WAP to read a string and replace the main string with the substring in the given position and find the length of the string before inserting and after inserting using string buffer.

SOURCE CODE

```
    import java.util.*;
    class stringbuffer{
    public static void main(String args[])
    {
    Scanner S= new Scanner (System.in);
    System.out.println("Enter sub string");
    StringBuffer sub = new StringBuffer(S.nextLine());
    System.out.println("Enter position to be inserted");
    int m=S.nextInt();
    System.out.println("Enteredstring:"+ms+"length:"+ms.length());
    ms.insert(m,sub);
    System.out.println("afterinserting:"+tms+"itslength:"+
    tms.length());
    }
}
```

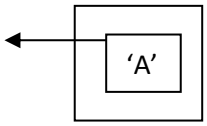
WRAPPER CLASSES

It is a class whose object wraps or contains primitive datatype.

- When we create an object to a wrapper class, it contains a field and in this field we can store primitive datatype.

  Wrapper class

Field



Why do we need wrapper classes?

- They convert primitive datatypes into objects and this is need on internet to communicate b/w two applications.
- The classes in java.util.*; package handle only objects and hence wrapper classes help in this case only.

Primitive values	wrapper classes
int	Integer
float	Float
double	Double
long	Long
character	Character

short	Short
boolean	Boolean
byte	Byte

Conversion from primitive to wrapper class object.

```
int a= 10;
Integer i = new Integer(a);
float f = 22.5;
Float f = new Float(f);
double d = 26.4;
Double D=new Double(d);
long l = 263453;
Long L=new Long(l);
char c = A;
Character C = new Character (c);
short s = 3;
Short s = new Shor(s);
boolean b = true;
Boolean b = new Boolean(b);
byte bt = 1;
Byte bt = new Byte(bt);
```

Conversion from wrapper class object to primitive datatypes :

```
int a = Int_dij.intValue( );
float f = Float_dij.floatValue( );
double d = Double_dij.doubleValue( );
long l = Long_dij.longValue( );
char c = Char_dij.charValue( );
short s = Short_dij.shortValue( );
boolean bl = Boolean_dij.booleanValue();
byte bt = Byte_obj.byteValue();
```

Conversion from primitive values to strings

```
String s1 = Integer.toString(10);
String s2 = Float.toString (25.5);
String s3 = Double.toString(25.4);
String s4 = Long.toString(26);
String s5 = Character.toString('a');
String s6 = Short.toString(12);
String s7 = Byte.toString(1);
String s8 = Boolean.toString(true);
```

Conversion from object/strings to objects/primitive values

```
ob1 / var 1 = Integer.ValueOf (string /object);
ob2 / var 2 = Float.ValueOf (string /object);
ob3 / var 3 = Double.ValueOf (string /object);
ob4 / var 4 = Long.ValueOf (string/object);
```

```
ob5 / var 5 = Character.ValueOf (string / object);
ob6 / var 6 = Short.ValueOf (string / object);
ob7 / var 7 = Byte.ValueOf (string / object);
ob8 / var 8 = Boolean.ValueOf /string / object));
```

Java program to connect primitive values into object wrapper class.

```
class wrapper
{
public static void main (String args[ ])
{

     int a = 100;
     float f = 23.4f;
     double d = 1234.56;
     long l = 985678;
     char ch = A;
     boolean bl = true;
     short sh = 128;
     byte bt = 12;
Integer obi = new Integer(a);
Float obf = new Float(f);
Double obd = new Double(d);
Long obl = new Long(l);
Short obs = new Short(sh);
Boolean obb = new Boolean(bl);
Character obc = new Character(ch);
Byte obbt = new Byte(bt);
System.out.println("Integer object value is "+obi);
System.out.println("Float object value is "+obt);
System.out.println("Double object value is "+obd);
System.out.println("Long object value is "+obl);
System.out.println("Short object value is "+obs);
System.out.println("Boolean object value is "+obb);
System.out.println("Character object value is "+obc);
System.out.println("Byte object value is "+obbt);

}
}
```

Java program to convert strings/object wrapper class into object/primitive values.

```
class wrapper2
{
Integer obi = new Integer(19850);
Float obf = Float(25.69f);
Double sh = 128;
Short obs = new Short(sh);
Boolean obb = new Boolean(false);
```

```
Character obc = new Character('a');
byte bt = 2;
Byte obbt = new Byte(bt);
int a = Integer.valueOf(obi);
float f = Float.valueOf (obf);
double d = Double.valueOf (obd);
long l = Long.valueOf(obl);
char ch = Character.valueOf (obc);
boolean bl= Boolean.valueOf (obb);
short s = Short.valueOf(obs);
byte b = Byte.valueOf (obbt);
System.out.println("integer primitive value is "+a);
System.out.println("float primitive value is "+f);
System.out.println("double primitive value is "+d);
System.out.println("long primitive value is "+l);
System.out.println("short primitive value is "+s);
System.out.println("boolean primitive value is "+bl);
System.out.println("character primitive value is "+ch);
System.out.println("byte primitive value is "+bt);
}
}
```

1) Java program to convert object wrapper class (number into primitive values(numbers)

```
class wrapperclass3
{
public static void main (String args[ ])
{
Integer obi = new Integer (19);
Float obf = new Float(25.69f);
Double obd = new Double(89567.23);
Long obl = new Long(987654321);
Boolean obd = new Boolean(false);
Character obc = new Character('a');
short sh = 128;
Short obs = new Short(sh);
byte bt = 2;
Byte obbt = new Byte(bt);
int a = obi.intValue(j);
float f = obf. floatValue);
double d=obd.doubleValue( );
long l = obl.longValue);
short s = obs.shortValue( );
byte b = obbt.byteValue( );
char ch = obc.charValue( );
boolean bl = obl.booleanValue( );
System.out.println("integer primitive value is "+a);
System.out.println("float primitive value is "+f);
System.out.println("double primitive value is "+d);
System.out.println("long primitive value is "+l);
System.out.println("short primitive value is "+s);
```

```
System.out.println("boolean primitive value is "+bl);
System.out.println("character primitive value is "+ch);
System.out.println("byte primitive value is "+b);
}
}
```

Java program to convert primitive/object wrapper class into strings.

```
class wrapperclass4
{
public static void main (String args [ ] )
{
int a = 100;
long l=9856578;
short sh = 128;
byte bt = 12;
float f = 23.4f
double d = 1234.56;
boolean bl = true;
String stra = Integer.toString(a);
String strl = Long.toString(l);
String strsh = Short.toString(sh);
String strbt = Byte.toString(bt);
String strsh = Double.toString(d);
String strlh = Character.toSring('c');
String strbl = Boolean.toString(bl);
System.out println("integer string value is "+stra);
System.out.println("long string value is "+strtl);
System.out.println("short string value is "strsh);
System.out.println("byte string value is "+strbt);
System.out.println("float string value is "+strf);
System.out.println("double string value is "+strsh);
System.out.println("charater string value is "+strlh);
System.out.println("boolean string value is "+strbl);
}
}
```

VECTORS

Vector is a pre-defined class which is used to create a generic dynamic array that holds objects of any data datatype and any number(Size). Vector is also called as generic dynamic arrays.

- The objects need not have homogenous data type.
- Vectors are created like arrays as shown below :

```
Vector obj = new Vector( );
Vector obj = new Vector(size);
```

Advantages of vectors

1) It is convenient to use vectors to store objects
2) A vector can be used to store a list of objects that may vary in size.
3) We can add & delete objects from the list whenever we require.

In vectors we cannot shore the primitive values directly, therefore we can 1st convert into object and then we can store in the list and this conversion can be done using wrapper classes.

- The vector class supports no. of methods that can be used to manipulate and some of the standard methods are:-

vobi.addElement (item); //add item specified to the list at the end position
vobi.elementAt (position); //returns the name of the given position.
vobj.size( );//returns the size of vector list.
vobi.removeElement (item);//removes the specified item from the list.
vobj.removeElementAt(position);//removes the item stored at the nth position
vobj.removesAllElements();//removes complete vector list.
vobi.copyInto(Array);//copies vector list to array list
vobi.insertElementAt (item,n);//inserts given item in given position.

- WAP to read and display details of a person like name, age, gender, weight, cell number using vectors

```
import java.util.*;
class vector_demo
{
Vector vob = new Vector( );
Scanner S = new Scanner(System.in);
System.out.println("enter ur name");
String name = S.nextLine( );
System.out.println("enter ur age");
Byte age = new Byte(S.nextByte( ));
System.out.println("enter ur gender");
Character gndr=new Character(S.next().charAt(0));
System.out.println("enter weight");
Float wt=new Float(S.nextFloat( ));
System.out.println("enter ur cell no");
Long cell = new Long(s.nextLong( ));
vob.addElement(name);
vob.addElement(age);
vob.addElement(gender);
vob.addElement(wt);
```

```
vob.addElement(cell);
System.out.println("given list is ");
for (int i=0;i<vob.size( ); i++)
{
System.out.println ("enter the new item to add");
System.out.println ("enter ur aadhar no");
Long adhar=new Long(S.nextLong ( ) );
System.out.println ("enter position");
byte pos=S.nextByte ( );
vob.insertElementAt (adhar,pos);
System.out.println("updated list is ");
for (int i=0;i<vob.size( );i++)
System.out.println(vob.elementAt(p));
System.out.println("enter it's pos to remove");
Int pos = S.nextInt( );
vob.removeElementAt(pos);
System.out.println ("updated list is:");
}
}
```