UNIT- III 05-02-19

INTRODUCTION TO JAVA

Java Vs C++

C++	Java
1. C++ is a Partially object oriented language.	1. Java is a fully object oriented language.
2.* It is a plat-form dependent language.	2. It is a Plat-form independent language.
3. The extension of C++ is .cpp	3. The extension of Java is .Java.
4. The Extension of output file is .Exe (Executable)	4. The Extension of output file is .class (Byte code)
5.* It is mainly used for system Programming.	5. It is mainly used for application Programming
6.* It will support Multiple Inheritance.	6. It doesn't support Multiple inheritance. but we can achieve the Multiple inheritance by using interface.
7)* In C++ we have Pointers	7) Java internally supports Pointers but we can't write Programing Pointers
8)* In C++ we have structures and unions	8) In Java their is no structures & unions
9)* In C++ we have call by value & call by refference.	9) In Java we have only call by Value.
10)* In C++ we have Virtual Keyword to override the functions	10) In Java their is no Virtual Keyword. Because all Nonstatic methods are override defaultly.

11) In C++ their is NO document section	11) In Java we have document section
12) C++ supports default arguments.	12) Java doesn't support for default arguments.
13) In C++ we have only * compiler thatway it is Platform dependent.	13) In Java it has both compiler & interpreter so that it is Platform independent.
14) * In C++ we can include the header files using include keyword.	14) In Java we can import the packages using import keyword
15) In C++ Every class Ends with semicolon.	15) In Java the semicolon is optional.
16) C++ is ~~not~~ a Case sensitive.	16) Java is a also Case sensitive language
17) In C++ the main method is defined after class.	17) In Java main method is defined inside the class.
18) * C++ doesn't have built-in libraries for threads.	18) Java has Built-in Packages for Threads.

06-02-19

Data types:-

Java has 8 Primitive data types. Those are byte, short, int, long, float, double, char, boolean.

→ The Primitive data types are also called as simple data types. All these Primitive data types are grouped in 4 groups those are integer, floating point, character, boolean.

Integers:-

Java has 4 integer data types. these are used to store the integer values the following table shows the integer data types.

Data type	bits	Range	Example
1. byte	8	-128 to 127	byte a;
2. short	16	-32,768 to 32,767	short b;
3. int	32	-2,147,483,648 to 2,147,483,647	int c;
4. long	64	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	long d;

Note:-
Java has only signed integer values those are -ve or +ve values. Theis is no unsigned integer values

floating point:-
Java has two floating point data types those are float & double. The float is used torepresent the single precision values, the double is used to double precision values. The following table shows the floating point data type.

Datatype	bits	Range	Example
float	32	$1.4e^{-045}$ to $3.4e^{+038}$	float f;
double	64	$1.9e^{-326}$ to 18^{e+308}	double e;

Character :-

Java has only one character data type i.e., char. It its not similar to C or C++ character data type. Generally in C or C++ it takes 1 byte (8 bits) of memory. but in Java 2 bytes of memory (16 bits). The following table shows the character data type.

Data type	bits	Range	Example
char	16	0 to 6,55,35	char g;

Boolean :-

Java has only one Boolean data type. generally it has only two values i.e., Either true or false (1 or 0). It has only 1 bit of memory.

Data type	bits	Range	Example
Boolean	1	true (or) false 1 or 0	boolean h;

Operators :-

Java has the following types of operators.

1) Arithmetic operators
2) Bitwise operators
3) Relational operators
4) Logical operators
5) Assignment operator
6) Conditional operator
7) Other operators.

Arithmetic operators:-

Arithmetic operators are used to Perform arithmetic operations. we have following arithmetic operators.

Operator	Name	Example
+	addition	a+b
−	Minus (Subtraction)	a-b
*	Multiplication	a*b
/	Divison	a/b
%	Modulus	a%b
++	increment	a++ (or) a+1
−−	decrement	a-- (or) a-1
+=	Arithmetic compound assignment Operators (or) short hand operators	a+=b (or) a=a+b
−=		a-=b
*=		a*=b
/=		a/=b
%=		a%=b.

Bitwise Operators:-

Bitwise operators are used to Perform Bitwise operations. These are applied on bits. The following are the bitwise Operators

Operator	Name	Example
~(tilled)	bitwise not	~a
&	bitwise AND	a&b
|	bitwise OR	a|b
^	bitwise X-OR	a^b
>>	shift right	a>>2

>>>	shift right with zero fill	a>>>24 (it is used to make a +ve num from the -ve)
<<	shift left	a<<2
&=		a=a&b
|=		a=a|b
^=	Bitwise compound	a=a^b
>>=	assignment operator	a=a>>b
<<=		a=a<<b
>>>=		a=a>>>b

Relational operators:-

Relational operators is used to between the two values. These are as follows

Operator	Name	Example
==	Equal to	a==b
!=	Not Equal	a!=b
>	greater than	a>b
>=	greater than are Equal to	a>=b
<	less than	a<b
<=	less than are Equal to	a<=b

logical operators:-

The logical operators are always used in b/w of two conditions. Internally Every condition has relation operator. These are as follows.

Operator	Name	Example
&	logical AND	(a>b)&(a>c)
|	logical OR	(a>b)|(a>c)
!	logical NOT	!(a>b)
&&	short circuit logical AND	(a>b)&&(a>c) [short ckt logical AND gets false when condition one sets false and No matter but condition set)
||	short ckt logical OR	(a>b)||(a>c) (short ckt logical of gets true and ~~false~~ first condition true no matter about condi -tion true)
==	logical Equal.	(a>b) = = (a>c)
!=	logical NOT Equal	(a>b) != (a>c)
?=	Conditional operator (or) if Else.	(a>b) ? a:b

Assignment Operator:

It is used to assign some value to the Variable

Operator	Name	Example
=	assignment	a=10; a=b=c=15;

Conditional Operator (?:)

Conditional operator is similar to if Else. it is also called as turnary operator. It has the following syntax.

Condition? Expression 1 : Expresion 2

→ As Per above Syntax whenever the condition sets to it will Execute the Expression 1, whenever the condition set false it will Execute the condition 2

$$\begin{tikzcd}\text{condition ?} \arrow[r, bend left, "\text{true}"] \arrow[rr, bend right, "\text{false}"'] & \text{Expression 1 :} & \text{Expression}\end{tikzcd}$$

Ex: int a=10;

((a%2)==0)? S.O.P (a+"is a Even") : SOP(a+"is a odd");

SOP = system.out.println

Other Operators:-

The other Operator has follows

Operator	Name	Example
( )	open Paranthesis/ Method operator	5+(4*2)
[ ]	square brackets/ array subscript operator	a[5];
.	member operator	obj.add();

Operator Precedence:-

Highest

()	[]	.	
++	--	~	!
*	/	%	
+	-		
>>	>>>	<<	
>	>=	<	<=
==	!=		
&			
^			
|			
&&			
||			
=	OP=		
lowest.			

Decision Making:-

Decision making statements are used to take the decision based on the condition. Every condition has two values those are true (or) false (1 or 0) we have the following decision making statements

i) if-Else
ii) Nested if
iii) if-Else-if ladder

Note:
All decision making statements are single selection statements.

If-Else:
If-Else statement is also called as Simple if. it has the following syntax.

```
if (condition)
{
// Expression 1;
}
Else
{
//Expression 2;
}
```

07-02-19

Example:-
write a Java Programm to check whether the given number is even or odd.

```
        class Simpleif
        {
        Public static void main(String args[])
        {
         int a=10;
         if((a%2)==0)
        {
      System.out.Println(a+"is even");
        }
       Else
       {
     System.out.Println(a+"is odd");
        }
    }
  }
```

Nested if:-

An if statement is with in another if statement is called nested if.

It has following syntax.

Syntax:-

```
if (condi)
{
if (cond2)
{
// stmt 1;
}
}
else
{
// stmt 2;
}
```

write a Java Programm to the maximum no among three numbers.

```
Class ~~Simple~~ Nested if
{
Public static void main(String args[])
{
int num1=10, num2=15, num3=12, max;
 if (num1 >num2)
  {
  if(num1 >num3)
   {
   max=num1;
   }
  else
  {
   max= num3;
  }
 Else
 if( num2 >num3)
  {
   max =num2;
  }
```

```
        Else
        {
         max = num3;
        }
      }
   System.out.Println("The Max is:"+ max");
    }
  }
```

If-Else-if ladder:

The if Else if ladder is used to check the conditio
whenever the first condition get false the control
will Enter in to the Else part and again it
check

if the second Condition is also false will enter.
into the one more Else if statement

Syntax:
```
        if (Cond1).
          {
           //stmt 1;
          }
         Else if (cond2)
          {
           // stmt 2;
          }
         Else if (cond3)
          {
           // stmt 3;
          }
         Else
           {
            // stmt 4;
           }
```

Write a Java Programm to find the Max num amon
Three numbers:

```
            Class ladder if
          {
          Public static void main (String args[])
```

```
{
int num1=10, num2=15, num3=12, max;
if (num1> num2 && num1>num3)
  {
   max= num1;
  }
 Else if (num2 >num3)
   {
   max= num2;
   }
  Else
   {
   max =num3;
   }
 system.out.Println ("The max is:"+max)
  }
}
```

(Extra concept)

Multi Selection Statement:

The multi selection stmts are used to take the decisions in multiple ways. The 'switch' stmt is a multi selection statement. It has the following Syntax

Syntax:

```
Switch (variable)
 {
 Case 0: //stmts;
         break;
 Case 1: //stmts;
         break;
 Case2 : //stmts;
         break;
      .
      .
      .
 default : //stmt4;
 }
```

Ex:-
write a Java Programm to Print week days using switch case.

```
~~Scoitch~~      class week
                 {
int ~~case~~ Public Static void main(string args[])
                 {
                 int w=5;
                 switch (w)
                 {
              Case 0 : S.O.P ("Monday");
                       break;
              Case 1 : S.O.P ("Tuesday");
                       break;
              Case 2 : S.O.P ("wednesday");
                       break;
              Case 3 : S.O.P ("Thursday");
                       break;
              Case 4 : S.O.P ("friday");
                       break;
              Case 5 : S.O.P ("Saturday");
                       break;
              Case 6 : S.O.P ("Sunday");
                       break;
              default : S.O.P ("invalid");
              }
         }
      }
```

Loops (Control statements or iterative statements)

loops statements are used to perform the iterations (or) repitions in Programming languages we have the following types of loops.

1) while
2) do-while
3) for

while:-

The while loop is a looping stmt. It has three Parts 1) initialization
2) condition
3) increment / decrement.

Syntax:

```
initialization;
while (condition)
 {
 //stmts;
 inc/Dec;
 }
```

write a Java Program to Print first 10 natural numbers using while

```
Class While.
{
Public static void main (String args[])
{
 int i=1;
 while (i<=10)
 {
System.out.Println("i is :"+ i);
  i++;
 }
 }
}
```

do-while:

do-while stmt is also looping statement. It is also having the same three Parts but the difference between while & do-while is that the do-while can Executes the statements ones even the condition gets false.

Syntax:-

```
initialization:
do
{
//stmts;
 inc/Dec;
} while (condition);
```

Write a Java Program to Print first 10 natural numbers using do-while loop?

```
        Class Do.while
          {
         Public Static Void main(String args[]);
          {
           int i=1;
          do
           {
           S.O.P ("i is:"+i);
           i++;
          } while (i<=10);
          }
      }
```

for:

for loop stmt is also a looping statement. It is also having three Parts. It has the following Syntax

```
    for(initialization; condition; inc/Dec)
       {
       //Stmts;
       }
```

Write a Java Programm to Print 100 natural numbers ~~using for loop~~ by Excluding Even numbers using for loop

```
      Class for
       {
    Public static Void main (string args[]);
       {
        int i;
      for(i=1; i<=100; i++)
       {
        if((i%2)!=0)
       {
       System.out.Println("i is:"+i);
    } } } }
```

for Each Version of for loop:-

It is another version of for loop. it will Pick the values from the array one by one. It has the following Syntax.

```
for(type var: array-name)
{
    // stmts;
}
```

Write a Java Programm to find the sumation of array Elements using forEach Version of for loop

```
class forEach
{
    Public static void main(String args[ ])
    {
     int num[ ] = {1, 2, 3, 4, 5};
      int sum=0;
     for(int x: num)
     {
       sum + = x;
     }
    System.out.Println("The sum is :" +sum);
    }
}
```

Branching:- (or) Jump statements

In Java we have 3 types of branching stmts Those are 1) break;

2) continue;

3) return

Scanned by CamScanner

Break:-

Break Statement is used to terminate the Process or control. Generally the break stmts are used in Multi selection stmt & looping stmts.

① write a Java Programm to print the first 5 natural numbers using break stmt

```
    Class Break;
    {
  Public static void main( String args[ ])
     {
      int i;
    for (i=1; i<=10; i++)
     {
       if (i>6)
      {
       break;
      }
    S.o.p ("The i value is :" +i),
     }
   } }
```

Ex:②

```
  Class Break
 {
 Public static Void main (string args[ ])
  {
   int i, j;
   for( i=0; i<3; i++)
    {
  S.o.Print(" Line: " +i),
   for ( j=0; j<100, j++)
    {
     if (j>10)
     {
     break;
     }
   S.o.p (" " +j);
    }
  } } }
```

Output

```
line0 0 1 2 3 4 5 6 7 8 9 10
line1 0 1 2 3 4 5 6 7 8 9 10
line2 0 1 2 3 4 5 6 7 8 9 10
```

→ The break stmt is also used along with the label names. generally it has the following syntax.

break label;

Example:

```
        Class Break.
        {
        Public static void main(String args[])
        {
        int i, j;
      Outer : for (i=0; i<3; i++)
        {
        system.out.Println ("line "+i),
        for (j=0 ; j<100 ; j++)
        {
        if (j>10)
        {
        break outer;
        }
        system.out.Print ("  "+j);
        }
        }
      }
    }
```

O/P:

```
line 0   0 1 2 3 4 5 6 7 8 9
10
```

Continue

The continue stmts are also used in looping stmt generally the continue stmt stop the current iteration and continue the next iteration.

The syntax as follows:

continue;

Ex:-

```
Class Continue.
{
Public static void main(String args[ ])
{
int i;
for (i=0; i<10; i++)
{
system.out.Print(" "+i);
    if ((i%2)==0)
    {
    continue;
    }
  system.out.Println(" ");
  }
 }
}
```

```
o/p
0 1
2 3
4 5
6 7
8 9
```

12-02-19

return:-

The return stmt is a another branching stmt. It Causes the Programm control transfer back to the caller of a method. The return statement is always the Exist statement of a method. The following Example, we are checking a number is Even or odd using check Even method. If the number is Even the method will return true other wise it will return false. So that the check Even method type is boolean.

Example:-
```
class return.
{
Public static Void main (string args[ ])
 {
 int num=10;
  check Even (num);
 }
}
Static ~~Void~~ boolean check Even (int n)
  {
   if ((n%2)==0)
   {
  system.out.Println (n+" is Even");
  } return true;
Else
{
System.out.Println (n+"is odd");
 return false;
 }
}
```

Classes and Objects:-

Class:-

A class is always refers behaviour & structure of an object in java we can create a class using "class" keyword. Every class has 'instance' variables and Methods. (or) A class is a blue print of an object.

Syntax:-

```
class class_Name
{
 type instance_var1;
 type instance_var2;
   |
   |
 type method1()
   {
   ----
   }
type method2()
   {___
   }
   |
type method n()
   {
   ----
   }
}
```

Example:-

```
class Box
{
int width;
int height;
int depth;
}
```

Object:-

An object is a runtime Entity of a class. it always refers the state of a class (or) the Properties of a class.

(or)

An object is an instance of a class. we can create the object using its class name and a "new" Keyword the new Keyword is used to allocate the memory to the object.

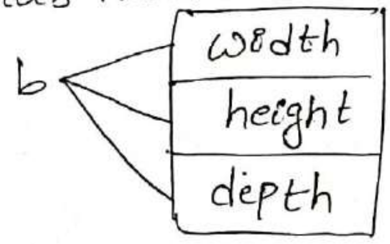
Syntax:

```
Class-name object =new class-name();
```

Example:-

```
Box b= new Box();
```

The above statement we have created an object 'b' of Box type. So that the object 'b' always refers the box class members.



Eg:- write a Java Programm to demonstrate the working of classes & objects.

```
Class Box
{
int width;
int height;
int depth;
}
Class Box Demo
{
Public static void main (String args[])
{
Box b = new Box ();
 b.width =10;
 b.height =20;
```

```
    b.depth =30;
  int volume = b.width * b.height * b.depth;
 System.out.Println("Box Volume is:"+ Volume);
 }
}
```

→ we can access the class members using its object name along with dot operator

Syntax:-

```
    Object. instance_Var1;
    b. width =10;
```

→ An object may refer another object.

```
        Box b1 = new Box();
        Box b2 = b1;
```

Eg②.

```
    class Box
    {
    int width;
    int height;
    int depth;
    }
  class BoxDemo
    {
  Public Static Void main(string args[])
    {
     Box b1 = new Box(),
     Box b2 = new Box();
      b1.width =10;
      b1.height=20;
      b1. depth =30;
  int vol1 = b1.width * b1.height * b1.depth;
  System.out.Println ("Box1 Volume is:"+ vol1);
   b2.width =1;
```

```
        b2. height=2;
        b2. depth =3;
    int vol2 =b2. width * b2. height * b2.depth;
    System. out. Println ("Box2 Volume is:" + vol2);
    }
  }
```

Methods:-

The method is a collection of statement that are grouped together to perform the specified operation or task. It has the following syntax.

Syntax:-

```
modifier return-type method-name (Parameter list)
    {
       //defination / body of the method;
    }
```

Modifier : It specifies three modifiers

1. public
2. Private
3. Protected.

return-type :- It specifies the return value of a method (int, char, boolean, double etc)

Method-name :- It specifies method name.

Parameter_list :- It specifies the arguments of the method.

Def (or) body :- It specifies the implementation of a method for Example

```
Public int add(int a, int b)
    {
    int c,
    c=a+b ;
    returne c;
    }
```

Eg:- write a Java Program to find the volume of a box by adding a method ?

```
     Class Box
    {
    int width;
    int height;
    int depth;
    Void volume()
   {
   int vol = width * height * depth;
  System.out.println("Box Volume is :"+vol);
   }
 }
  class Box Demo
   {
 Public static void main(String args[])
   {
   Box b=new Box().
     b.width =10;
     b.height =11;
     b.depth = 12;
     b.volume();
     }
   }
```

Scanned by CamScanner

Return a method Volume:-

A method Can return a Value of a return type generally we use the return statement at the End of a method.

Eg:
```
Class Box
{
int width;
int height;
int depth;
int volume()
{
return width * height * depth;
}
}
Class BoxDemo
{
Public static void main (string args[])
{
Box b= new Box();
b.width =10;
b.heeght =11;
b.depth =12;
int vol =b.volume();
system.out.Println ("Box Volume is :" + Vol)
}
}
```

Eg②:-
```
Class Box
{
int width;
int height;
int depth;
Void set Dim (int w, int h, int d)
{
width =w;
height=h;
```

```
        depth = d;
      }
      int Volume ()
     {
     return width * height * depth;
      }
   }
   class Box Demo
    {
   Public Static Void main(string args[ ])
   {
  Box b= new Box();
   b.set Dim(10,11,12);
   int vol = b.volume ();
System. out. Println ("Box Volume is:" + Vol);
    }
 }
```

Eg: ③

```
      class Box
       {
      int width;
      int height;
      int depth;
     int Volume (int w, int h, int d)
      {
        width = w;
        height = h;
        depth = d;
      return width * height * depth
       }
   }
    class Box Demo
     {
    public Static void main (String args [ ])
    {
   Box b1 = new Box();
```

```
Box b2 = new Box();
int vol1 = b1.Volume (10,11,12);
system.out.Println ("Box1 Volume is:" +vol1);
   int vol2 = b2.Volume(1,2,3);
 System.out.Println("Box2 Volume is:"+vol2);
 }
}
```

Extra concepts:-

Constructor:-

A constructor is a special member of a class whic
is used to initialize the instance variables automatica
The constructor name is same as class name. It
doesn't have any return type because internally. It
is a class type a constructor can be invoke automati
cally wheneven an object is created. We have two
types of constructors.

1. Default constructor
2. Parameterised constructor.

Default Constructor:-

Constructor which doesn't have arguments (or)
Parameters is called Default Constructor. It has
the following syntax.

Syntax:-

```
class class-name
{
 class-name()
 {
  - - -
 }
}
```

Ex:

```
Class Box
{
Box()
{
- - -
}
}
```

Parameterised Constructor:-

A constructor which has Parameters is called Parameterised Constructor we can pass local Variables as a Parameter and as well object as a Parameter (Object as a Parameter - copy Constructor)

Syntax:
```
class class-name
{
clas-name(Parameter list)
  {
  - - - -
  }
}
```
Example.
```
class BOX
{
BOX(int w, int h, int d)
{
- - -
}
}
```

Garbage collection:-
(Extra)

In C++ we can deallocate the memory using Delete operator and we can destruct the object using destructor, but in Java the memory is deallocated automatically such type of deallocation is called Garbage collection.

The Java deallocates the memory whenever the object doesn't have any reference. So that those objects are no longer to be use.

Eg:- write a Java Programm to demonstrate the working of Constructor in Java.
```
class Box
{
int width;
int height;
int depth;
Box()
```

```
{
system.out.Println("default Constructor");
    width=10;
    height=11;
    depth=12;
    }
  Box(int w, int h, int d)
  {
 System.out.Println("Parameterised constructor");
    width=w;
    height=h;
    depth=d;
    }
  Box(Box b)
  {
  system.out.Println("object as a Parameter");
  width=b.width;
  height=b.height;
  depth=b.depth;
  }
  int Volume()
  {
 return width* height * depth;
  }
}
 Class BoxDemo
  {
 Public Static Void main(Strings args[])
  {
  Box b1=newBox().
  Box b2 = new Box(1,2,3);
  Box b3 =new Box(b1);
```

Two dimensional arrays:-

In 2-D array we have two dimensions 1-D refers rows, 2-D refers columns. we can create a 2-D array using following syntax.

Syntax

Type array name[ ][ ]= new type [row] [col];

(or)

Type [ ][ ] array name = new type[row][column];

Example:

int a[ ][ ] = new int[4][5];

(Or)

int[ ][ ] a= new int [4][5]:

write a Java Program to demonstrate the working of 2-D array

```
class TwoD
{
  Public static void main(string args[ ])
  {
    int a[ ][ ] =new int [4][5];
     int k=1;
    for(int i=0; i<4; i++)
     {
     for (int j=0; j<5; j++)
      {
       a[i][j] = k;
         k++;
     } }
  System.out.Println ("The array elements are:");
   for (i=0; i<4; i++)
{
for(j=0 ; j<5; j++)
{
System. out. Print ("  "+a[i][j]);
}
System. out. Println ("  ");
}
}
}
```

Scope:

The scope always determine which objects are visible to others it can also determines the life time of a variables. Generally the scope will be start whenever the block will be start. the scope will be end whenever the block will be end.

→ The Block will be ~~case~~ opened using open curly brace ({) the block will be closed using closed curly brase (}) ..

→ In java Programm we can create any no.of blocks.

write a java Programm to demonstrate the working of scope.

```
Class Scope
{
public static void main (string args[]),
{
 int x=1;
 if (x>=1)
 {
  int y=20;
 System. out. Println (" "+x+" and"+y);
  x=y*2;
 }
} S.O.P ("x is :"+x);
```

Eg② :- write a Java Program to demonstrate the working of a life time of a Variable.

```
O/P        class life time
           {
           Public static void main(string args[])
           {
           for(int i=0; i<3; i++)
-1         {
100          int Y=-1;
-1         System.out.Println("Y is :"+Y);
100            Y=100;
-1         System.out.Println ("Y is :"+Y);
100          }
            }
           }
```

20-01-19

String Handlings:-

String is a sequence of characters in Java, a string is an object which refers the sequence of characters to create a string in Java we have the following three classes

1) String
2) String buffer.
3) String Builder

In Java strings are immutable (do not change) but the String Buffer & String Builder classes are mutable (we can change) because the modifications can be happen on string class. If you want to modified the string you have to keep the original string in another string then we can have to modified it.

In Java we have two different ways in order to Create strings.

1. By using literals (" ")
2. By using "new" operator/Keyword.

By using literals:-

we can create the strings using literals as follows

```
String str = "Hello";
```

where String is a class name & str is a instance of String class.

→ It is always refers "Hello" strings.

By using 'new' Keyword:

we can also create the String using "new" operator. Here we will always call the string class constructor. It has the following different forms.

1. String str = new String();
2. String str = new String ("AITS");
3. String st = new String (String obj);

$$\begin{tikzcd} str \arrow[r] & \boxed{AITS} \\ & st \arrow[u] \end{tikzcd}$$

Eg:- String st = new String (str);

4. String st = new String (char[]);

Eg:- char c[] = {'W', 'O', 'R', 'L', 'D'};

String st = new String (c);

5. String st = new String (char[], int start, int numchars);

char[] - character array

start - starting index of char array

numchars - The no. of characters loaded in a String.

Eg:- String st = new String (C, 0, 3); //WOR

6. string st = new string (byte [ ]);

Eg:- byte[ ] b = { 65, 66, 67, 68 };

string st = new string (b); //ABCD.

7. string st = new string (byte [ ], int start,
int num bytes);

Eg:- String st = new string (b, 0, 2); //AB.

Programm:

```
class String Demo
{
Public static Void main( string args[ ])
{
string str1 = "Helloo";
String str2 = new string (str1);
String str3 = new string ("welcome");
char ch[ ] = {'A', 'I', 'T', 'S'};
string str4 = new string (ch);
System.out.Println (str1);
system.out.Println (str2);
System.out.Println (str3);
System.out.Println (str4);
}
}
```

```
O/P:- Helloo
      HELLO
      welcome
      AITS
```

String handling functions:

1) length( );

length( ) is used to find the length of a string. Suppose if you want to find a particular string length then we can invoke the length function using the

Same String Object.
It has the following syntax

Syntax: int length()

Eg:-

```
Class StringLength
{
Public static void main (string args[])
{
string str1 = "HEllo";
String str2 = newString (str1);
String str3 = new String ("WELCOME");
char ch[] = {'A', 'I', 'T', 'S'};
string str4 = new string (ch);
S.O.P (str1);
S.O.P (str2);
S.O.P (str3);
S.O.P (str4);
S.O.P (str1. length());
S.O.P (str2. length());
S.O.P (str3. length());
S.O.P (str4. length ());
}
}
```

o/p

```
Hello
Hello
welcome
AITS
5
5
7
4
```

2) String Concatenation:-

In Java we can Concatenate two strings using '+' operator and also we can concatenate the two strings using 'Concat()' method.

Syntax:-

String Concat (String obj);

Eg:- Str₁ = "Hai";
Str2 = "How are you";
→ Str₁ . concat (str2);
→ Str2 . Concat (str₁);

22-02-19

→ In the above Example Str₁ . concat (str2) str2 is a argument String and str₁ is a invoked string. So that we are concatenating both str₁ & str2 i.e., the argument string is concatenated to invoked string So that theoutput will be HaiHow are you.

→ we can also perform concatenation using '+' operator in b/w of two strigs for Example Str2 + str₁
O/P is How are you-Hai

Eg:- ①

```
Class ConcatDemo
{
Public static void main (string args [])
{
int age = 9;
String str = "He is" + age + "Years old";
System . out . Println (str);
}
}
```

O/P: He is a Years old

②

```
Class ConcatDemo
{
Public static void main (stringargs[])
{
System. out. Prinln ("The result is:" + 2+2);
}
}
```

O/P :- The result is : 22

Note:-
suppose System. out. Println ("The result is:"+(2+2));
O/P ⇒ The result is: 4

String Conversion:-

In Java we have the string conversions i.e., from any data type to string type so the ValueOf() method is used to convert any data type variable to string type but if we want to convert the object to string type we have to use toString() method.

Syntax:-

```
String valueOf (type var);
              ↳ any type
String toString ();
```

Write a Java Program to demonstrate the working of ValueOf() method.

```
Class Value Of
{
 Public static void main (String args [])
 {
    int a = 10;
  System. out. Println (a+90); //100
  String str = String. ValueOf (a);
  System. out. Println (str+90); // 1090
 }
}
```

write a Java Program to demonstrate the working of toString () method.

```
Class Student
{
String name;
int id;
Student (String name, int id)
```

```
    {
    This. name = name;
    This . id = id;
      }
    public String toString ()
      {
    return "     " +name+ "     " +id;
    }
  }
    Class ToString
   {
   Public static void main (string args[])
   {
  Student S1= new student ("EEE", 1241);
  Student S2 = new student ("IT", 201);
   System. out. Println (S1);
   S.O.P (S2);
  }
 }
```

Note:-
The tostring() method describes the object. The Java automatically call tostring() method when Ever we are going to display object.

Character Extraction:-

Using character Extraction method we can Extract the characters from the string. we have the following character Extraction Methods

i) CharAt():-

The charAt() method is used to Extract a Particular Character from the string based on the specified location.

Syntax:

```
char charAt (int loc);
```

Scanned by CamScanner

Program:-
```
  class CharAt
   {
  public static void main (string args[])
    {
    String str = "HELLO";
    System.out.Println (str.charAt(1));
    }
                         O/P:- E
  }
```
ii) get Chars ():-

A getchars() method is used to extract the multiple characters from the string.

Syntax:-
```
  void getChars(int start, int end, char target[],
                              int target start);
```
Program:-
```
  class GetChars
  {
  Public static void main (String args[])
  {
                 0 1 2 3 4 5 6 7 8 9 10 11 12
   String str = "This is AITs college";
   int start = 8;
   int end = 12;
   char buf[] = new char [end - star];
   str.getChars (start, end, buf, 0);
     System.out.println (buf);
   }
                     O/P:- AITs
  }
```
iii) get Bytes ():-

A getBytes () is similar to getChars () but it will Extract the bytes of a character in a string

Syntax:-
```
  byte[] getBytes ();
```

Program:

```
    Class GetBytes
    {
    Public static void main (String args[])
      {
    String str = "This is AITS college";
      byte[] b = str.getBytes();
     for(int i=0; i< b.length; i++)
     System.out.Println ("   " + b[i]);
    }
   }
```

iv) toCharArray():

It is used to convert the given string into array

Syntax:

```
char[] toCharArray();
```

Ex:

```
    Class TocharArray
    {
    Public static void main(String args[])
      {
    String Str= "This is AITS college",
     char c[] = str.toCharArray();
      System.out.Println (c[8]);
     }
    }
```

String Comparisions:

Two Compare the Strings we have the following methods.

i) Equals()

ii) equalsIgnoreCase()

iii) regionMatches()

iv) Starts With ();

v) Ends With ();

vi) compare To ();

i) equals():-

The Equals method will compare the two strings if the both strings are Equal it will return true otherwise false.

Syntax:-

boolean equals (String str);

Program:-

```
Class Equals{
{
Public static void main(String args[])
{
String  str1 = "hello";
String  str2 = "hello";
string  str3 = "EEE";
string  str4 = "Hello";
S.O.P(Str1.Equals(str2)); → True
S.O.P (Str1.Equals (str4)); → false
S.O.P (str4.Equals (str3)); → false.
}
}
```

ii) Equals Ignore Case():

It is Same as equals method but it ignores the case.

Syntax:-

boolean equalsIgnoreCase(String Str);

Program:-

In the above Example Str1 & str4 are different when we use Equals Method. But these are Same when we use EqualsIgnore Case method. Because it is ignoring the case.

```
S.O.P(Str1.EqualsIgnoreCase (str4)); → True
```

iii) regionMatches() :-

regionMatches is used to match the particular regions in a specified strings.

Syntax:-

boolean regionMatches(int start, String str2, int str2 index, int numchars);

start - starting index of a $\overset{(or)}{\text{start}}$ invoked string (str1)

Str2 - Second String

Str2 start - starting index of a str2.

numchars - no.of characters from string to starting index.

Program:-

```
class Region
{
Public static void main(String args[])
{
String str1 = "This is AITS college";
string str2 = "welcome AITS";
  S.O.P(str1. region Matches(8, str2, 8, 4));
}
}        O/P:- True
```

23-02-19

iv) startsWith():-

It is used to check wheather the string is started with specified or not. If it is started it will return $\overset{\text{True}}{\wedge}$ otherwise false

Syntax:-

boolean startsWith(string str);

boolean startsWith(string str, int loc);

Program:
```
        class StartsWith
        {
        Public static void main (String args[])
          {
        String str = "welcome to AITS";
True //System. out. Println (Str. startsWith ("weL") ;
True //system. out. Println (str. startsWith ("to", 8));
        }
    }
```

v) EndsWith():-

It is used to check whether the string is End by the specified String or not. It can also coretlen true on succes otherwise false.

Syntax:-

boolean EndsWith (String str);

Program:
```
      Class EndsWith
      {
      Public static void main (String args[])
      {
      String str = "welcome to AITS";
        S.O.P (str. EndsWith ("AITS")); → True
        S.O.P (str. EndsWith ("wel")); → false.
      }
    }
```

vi) CompareTo():-
(Imp)

CompareTo method is used to Perform the Comparision b/w the strings. generally Equals method Compare the Strings but not sorting if we want to sort the strings in Dictionary mannar we can Prefer compareTo Method.

Syntax:-

int CompareTo (String str);

It will return the following Values

lessthan zero — invoked string is less than str

Greaterthan zero — invoked string is > str

Equalto zero — invoked string is same as str.

Ex:-

```
class CompareTo
{
Public static void main (String args[])
{
String str[] ={"This", "is", "AITS", "college"};
for(int i=0; i<st.length ;i++)
{
for (int j= i+1 ; j<st.length ;j++)
{
if(st[j].CompareTo(st[i])<0)
{
String temp =st[i];
st[i] =st[j];
st[j] =Temp;
}
} }
s.o.p ("The sorted strings are:");
for (int k=0; k<st.length ; k++)
s.o.p("  "+st[k]);
}
}
```

Searching a string :-

The String searching methods are used to search a particular character or substring in a string. we have the following two methods.

i) indexOf();

ii) lastIndexOf()

indexOf():-

It is used to search first occurances of either character or substring.

Syntax:-

1) int indexOf (char ch);
2) int indexOf(char ch, int index);
3) int indexOf (String str)
4) int index Of(String str, int index)

one success it will written +ve values otherwise it will written '-1'.

1. 1st form of method will search the 1st occurance of a character from the starting location.
2. 2nd form of method will search the character from the specified index location
3. 3rd form of method will search the string from the starting index.
4. 4th form of method will search the String from the specified index location.

lastIndexOf():

It is used to search last occurances of a character or sub string.

Syntax:

```
int lastIndexOf(char ch);
int lastIndexOf(~~String~~ char ch, int index);
int lastIndexOf(String str);
int lastIndexOf(String str, int index);
```

On success it will written +ve values otherwise it will written -1.

Program:-

```
Class Search
{
Public static void main (String args[])
{
String str = "This is AITs college";
S.O.P(str. indexOf('T'));
S.O.P(str. indexOf('T', 5));
S.O.P (str. indexOf("is"));
S.O.P (str. indexOf("is", 5));
S.O.P(str. lastIndexOf('T'));
S.O.P (str. lastIndexOf('T', 5));
S.O.P (str. lastIndexOf("is"));
SO.P (str. lastIndexOf("is", 5));
}
}
```

Modifying Strings

We can Modify the string using following methods. generally we can't modify the Strings because these are ~~rebuitable~~ Immutable.

Scanned by CamScanner

The following methods are used for modifying the string but the same modification is not Effected on Original String.

25-02-19

The following methods are used to Modifying a String

1. Substring ( )
2. Concat ( ) (already discuss)
3. replace ( )
4. trim ( )

Substring ( ):-

Substring method is used to extract the part of the string from the original string.

Syntax:- String substring (int index);
String substring (int startindex, int Endindex);

Eg:-
```
Class SubstringDemo
{
Public static void main (String args[])
{
String str="This is AITS college";
System.out.Println(str.substring(8)); //AITS college
System.out.Println(str.substring(8,11)); //AITS
}
}
```

replace ( ):-

replace method is used to replace char ~~array~~ or a string

syntax:-
String replace (char ch_1, char ch_2);
String replace (CharSequence str_1, CharSequence str_2);

Ex:-
```
Class ReplaceDemo
{
Public static void main (String args[])
{
String Str="This is AITS College";
System.out.Println (str.replace ('+', 't'));
System.out.Println (str.replace ("is", "was"));
}
}
```

trim ():-

It is used to remove the leading (starting) & tailing (Ending) ~~ca~~ white spaces of a String.

Syntax:-

```
String trim()
```

Eg:-

```
Class TrimDemo
{
Public static void main (String args[]);
{
String ~~S~~tr = "  welcome  ";
System.out.Println (Str.trim());
}
}
```

Other Strings.

isEmpty ():-

To check ~~w~~heather the string is Empty or not. on success it will written true otherwise false

Syntax boolean isEmpty()

Split:- (Imp)

It is used to split the entire string into individual strings based on the regular given Expression

Syntax:- String[] split (String regExp);

Eg:-

```
Class SplitDemo
{
Public Static Void main (String args[]).
{
String str="This is AITS College";
```

```
    String st [] = str.split(" ");
    for (int i=0; i<st.length; i++).
    s.o.p (st [i]);
   }
  }
```

Contains ( ):

It is used to check the specified string is in original String or not.

boolean Contains (String str);

Ex:-

```
    Class Contain
    {
   public static void main(String args[])
   {
   string Str = "This is AITS College";
    s.o.p (str. contains ("ALTS")); // False.
    }
  }
```

String Buffer ( ): // Extra topic

String class is a fixed length character sequence So that the strings are immutable but String Buffer class is a growable and writable Character Sequence so that it is a mutable.

for Every string it has the addition 16 space other than the String length.

→ we have the following string buffer constructors!

1) String Buffer( );
2) String Buffer(int size);
3) String Buffer (String str);
4) String Buffer (CharSequence str);

Methods:-

1. int length():- used to find the length of string buffer

2. int Capacity():- used to find the capacity of string buffer.

3. Void Setlength(int):- used to set length of a string Buffer

4. Void EnsureCapacity(int):- used to increase the capacity of a String Buffer,

5. int CharAt (int) :- it is used to find a Particular character in a specified location.

6. Void ~~s~~et CharAt(int):- used to set character at a Particular location.

7. Void getChars (int, int, char[], int):-
It is used to Extract Multiple characters from a string.

8) StringBuffer append (int); (add) used to append the number to the original.

9) StringBuffer append (string):- used to append the string to the Original

10) StringBuffer insert (int loc, char ch);
11) StringBuffer insert (int loc, char string); } It is used to insert Either a char or a string at specified location.

Ex:
```
Class Insert
{
Public static void main(String args[])
{
StringBuffer sb=new StringBuffer ("Hello AITs");
    S.O.p (sb); // Hello AITs
    S.O.p (sb.insert (6, "This is"));
    S.O.P (sb); // Hellow this is AITs.
}
}
```

12) String^Buffer delete (int start, int End);
It is used to delete the string from specified starting to Ending index.

Ex: S.o.P(sb. delete(0,4)); // This is AITS

13) String^Buffer deleteCharAt(int loc);
It is used to delete a particular character at specified location.

Ex: S.o.b (sb. deleteCharAt(0)); // his is AITS

14) String^Buffer replace (int start, int End, String str);
It is used to replace a specified string with another string.

Ex: S.o.p (sb. replace (0, 3, "This")); // This is AITS

15) StringBuffer substring (int index);

16) String Buffer substring (int start, int end):

It is used to Extract the substring from the original string.

17) StringBuffer . reverse():
It is used to reverse the original string.

Example:

```
Class StringBufferDemo
{
 Public static void main (string args[])
 {
 StringBuffer sb1 = new StringBuffer("Hello AITS");
 StringBuffer sb2 = new StringBuffer ();
  S.O.P (sb1. length()); // 10
  S.O.P (sb2 . length ()); // 0
  Sb2 . Set length (5); ~~//5~~
   S.O.P (sb2. length ()); // 5
  S.O.P (sb1. capacity()); // 26
  S.O.P (sb2. Capacity()); // 21
```

Scanned by CamScanner

```
Sb2 . EnsureCapacity(40); //
S.o.p(sb2.capacity()); // 40
S.o.p(sb1.charAt(2)); // L
S.o.p(sb1.setCharAt(2,'A')); HEALO AITs
S.o.p(sb1);
S.o.p(sb2.substring(6)); // AITS
S.o.p(sb1.substring(7,8)); // IT
}
}
```

String Builder:-

It is similar to String Buffer class but it is not safe for threads. if we are using threads in our program it is better to consider StringBuffer class instead of StringBuilder class.

Assignment Questions.

1. a) Differentiate b/w Java & C++?
   b) list the various data types in Java Explain with Suitable Example?

2. Define classes & objects in Java & Explain with suitable Examples.
   b) Explain Various String handling functions in Java?