27/11/018

unit-1

⇒ Introduction to object oriented programming language.

→ oop language based on the concept of object and classes.

→ And object contents data in the form of fields, and the care that can be represented in the class, with the help of methods. for example oop are C++, JAVA PHP, Ruby, pesl, pascal, and lisp programming language.

→ The major modivative factor of an oop is to elimi-nate some of the problems, that are occur in procedural oriented programming language (POP).

→ object oriented programming language treated data is a critical in the development of a program.

→ Some of the points of oop language are:—

① oop concentrate on the data, whether than procedure.

② programs are divided into in the form of objects.

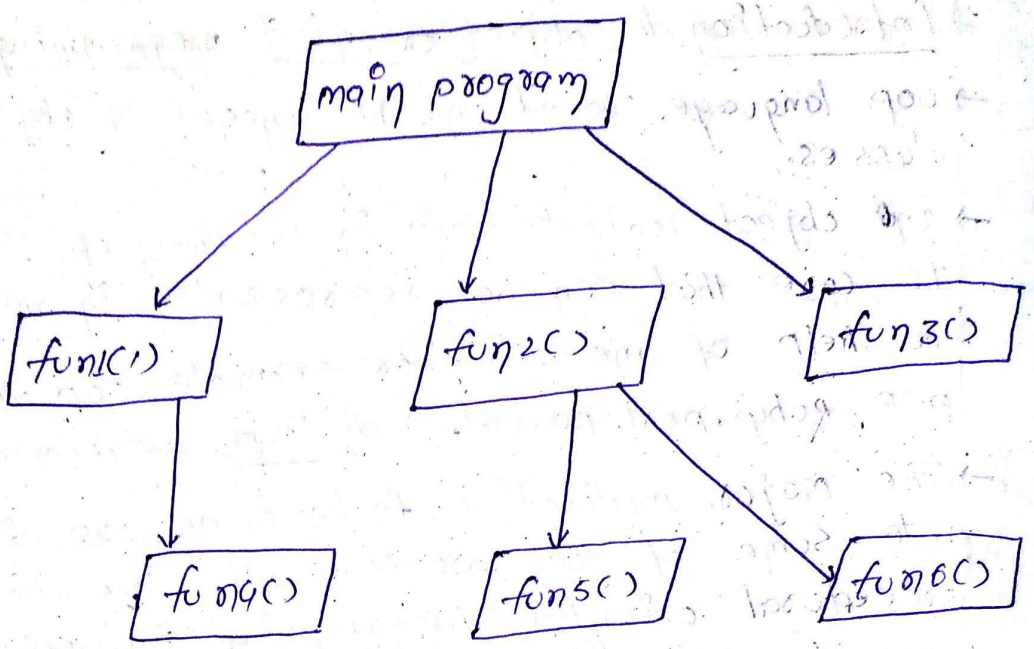
③ objects can communicate with each other with the help of methods.

⇒ POP language

→ Convential programming languages using high level language such as C, cobol and fortran are commonly known as pop language.

→ In procedural oriented programming language the problem is divided into number of function and then after solving, these functions we can get the result of main program.

→ This can be represented as:-



→ pop language contents a list of instructions for the computer and organize these instructions into groups known as functions.

→ we normally use a flow chart to organise these instruction and represents the flow of control from one action to another.

→ In pop language, we are giving the less important to data.

⇒ <u>principles of oop languages</u>

The basic principle of oop language are:-

① <u>class</u>:-

→ A class is a model for creating an object ⇒ a class is a combination of some data member and member functions.

→ The syntax for creating a class is-

```
class class name
{
  data members;
  member function;
}
```

→ In the above syntax class is a keyword followed by the name of the class and in between opening and closing braces, we will declare some data members and member function.

→ By using the above syntax, we can represents a class as-

```
class ss
{
  int a,b;
  void display();
}
```

→ In the above ~~class~~ code, we declare a class, with a name ss, in that class, we declare some variables with an integer data type, known as data members and we declare a function display known as member function.

② object :-

→ An object is an instance of a class, by using the object we can call the methods that are declare in a class.

→ The syntax for creating an object is-

```
classname objectname = new class name();
```

by using the above syntax, we can create an object for a class ss is —

```
ss k = new ssc();
```

→ In the above statement we create an object k for a class ss and the new operator is used to allocate sufficient memory for an object k.

③ Data Encapsulation and data abstraction :-

→ The wrapping (or) combining of a data and function into a single unit is known as data encapsulation

→ A class is an example of data encapsulation.

→ Data abstraction refers to act of representing the accential feature without including the background details of a program.

④ Inheritence :-

→ The process of creating a new class from already existing class is known as inheritence.

→ The existing class can be known as super class (or) base class (or) parent class.

→ The Newly created class is known as sub class or derived class (or) child class.

→ The inheritence concept provides the idea of reusebility and it also used for reducing the size of a program.

→ Reausebility means without declaring the data of a parent class, we can use the data from a child class.

⑤ polymorphism :-

→ In polymorphism - poly means many and morphism means forms. polymorphism means an ability to do more than one task is known as polymorphism

→ An operator overloading can be consider as an example of polymorphism.

⑥ Dynamic binding :-

→ Dynamic binding refers to the linking of a procedure call to the code that will be

executed at runtime only. The code is associated with the procedure. is not know untill the program is executed.

→ The dynamic binding can also be known as late binding.

⑦ message passing :-

→ message passing is the process of sending and receiving the information.

→ In oop the information can be transformed into a program by using objects.

⇒ Application of oop

→ The applications of oop can be classified into four types :-

① stand alone application :-

→ These application can be known as window based application. These are the application, that we need to instant into a every computer such as mediaplayer and antivirus.

② web application :-

→ An application that we run at server site is known as web application. currently we are using higher text marcor language (HTML), Servers, Java Seach page (JSP), strub technology can be used for designing web application.

③ Interprice application :-

→ The banking applications are come under the inter price application.

→ These applications can be developed by using Enterprice. Javabeans (EJB)

④ mobile application :-

→ An application that can be created for a mobile device is known as mobile application.

→ The mobile applications can be desigg by using Java me (Java mobile enterprice).

⇒ History of Java

→ Java is a pure oop language, means in Java we can write a program by using the class and. without using a class, it is not possible to write a program in Java.

→ Because of this reason, we are saying Java is a pure object oriented programming languages.

→ In January 1991, MIK-scheriden, pattric newghtoon and James-Gausling are started a language which is more simple in easy way for developing the programs.

→ mik-scheriden first focus on business development application and pattric newghton first focuson graphical system and James-Gausling was first identify a proper programming language for deriving purpose.

→ At first they name the programming language-are OAK (or) later on 1995 they change the name of the programming language OAK to JAVA.

→ while designing the programming language, they consume, a lot of tea. and they believe that the tea is also played an important role while developing the Java programming language, because of this reason. They design the symbol of Java P.L as cup and soce.

⇒ Feature of Java (or) Java Buzzword :-

following are the feature of Java :-

1. simple
2. object oriented
3. distributed
4. Robust
5. system independence
6. Interpreted
7. multithread
8. scalability

(1) simple :-

Java is a simple programming language for making simple. some of the difficult concepts of c and c++ such as pointer are completely eliminated from Java and they follow same syntax of c and c++ for designing a program.

(2) object oriented :-

→ Java is a pure object-oriented programming language.

→ A Java program can be created by using classes and object.

→ A class is a model for creating an object can be used for calling the methods that are declare in a class.

(3) distributed :-

By using the Java program we can transform the information onto the claim, this is possible because of Java can handle the proto calls like-transmission control protocall (TCP) and user data gram protocal (UDP).

④ Robust :-

→ Robust means strong.

→ Java programs are strong because they do not crases easily like c (or) c++ program.

→ Java has an exclient in-built exception handling mechanism.

→ An exception is a run-time error.

⑤ System indepence :-

Java is a system independent language, we can compile and run the Java program in any computer where the Java software has been installed.

⑥ Interpreted :-

→ c and c++ can be used either compiler (or) interpreted to execute the program.

→ Java can be used both compiles and interpreter to execting the program.

⑦ multithread :-

A thread represents a single process multiple threads running on multiple process is known as multithread.

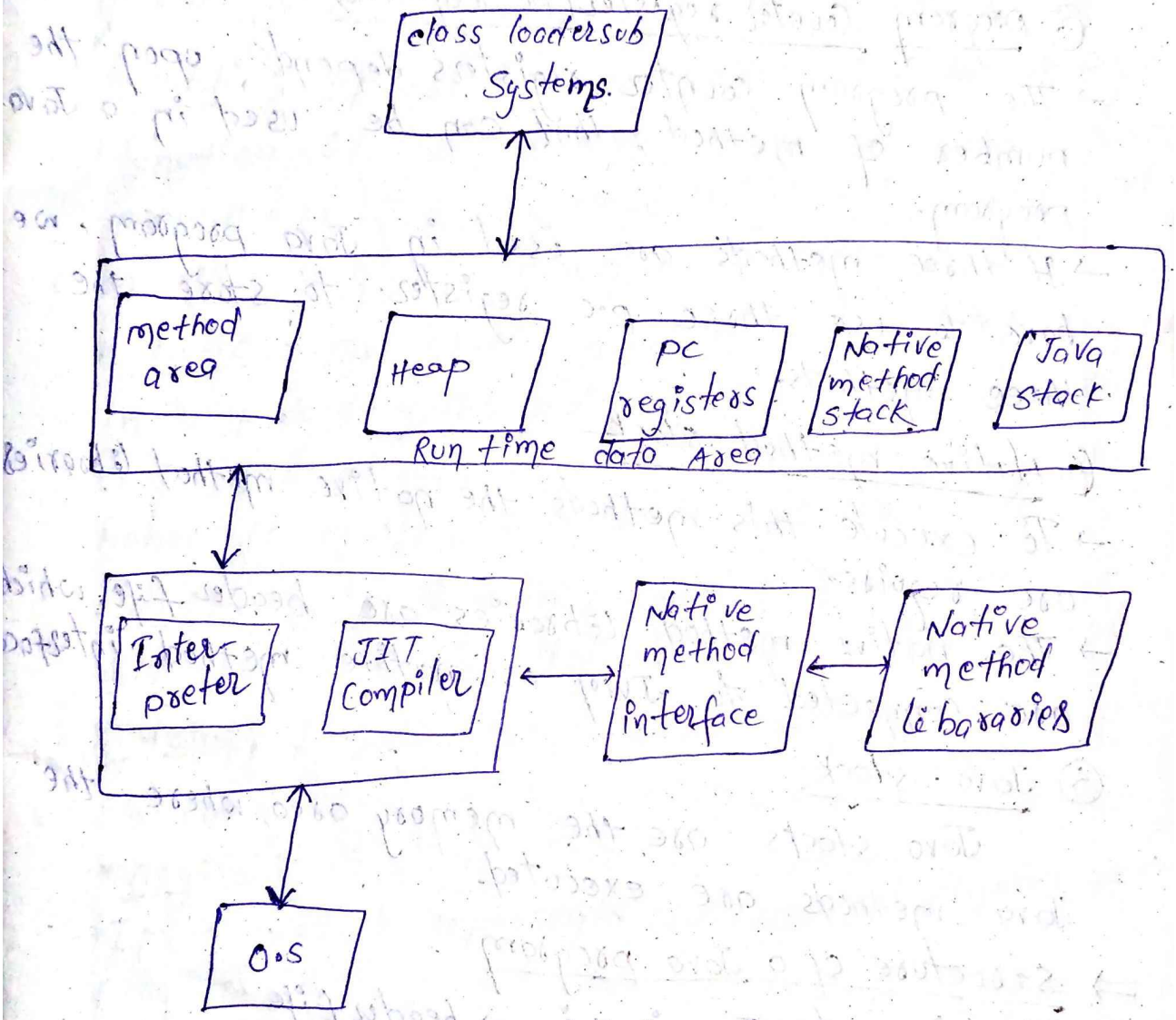
⑧ scalability :-

Java can be implemented on wide range of compoter application, with varying labour of resoureces. staoting from embaded systems to main frame computer.

⇒ Java virtual machine (JVM)

→ The Java virtual machine is the heart of entire Java program execution.

→ The structure of Java virtual machine can be represented as:-



class loader sub systems:-

→ The .Java file is converted into .class file

→ The .class file is loaded into the class loader Sub systems which contents Java byte code.

→ Then it varifies all the byte code instruction in that .class file are property or not.

→ If the byte code instructions are proper then, the class loader subsystem allocates sufficient memory for the program execution.

→ The memory area of Java virtual machine can be divided into five types.

① method area :-

It is a block memory, which stores class code, code of the variables and code of the methods.

② Heap :-

The heap can be used for storing an object of a class.

③ program counter register (PC Register)

→ The program counter registers depends upon the number of method, that can be used in a Java program.

→ If three methods are used in Java program, we have to use three P.C register to store the three methods.

④ Native method stack

→ To execute this methods the native method libraries are require.

→ The native method libraries are header file, which are connected to JVM as native method interface.

⑤ Java stack

Java stacks are the memory area, where the Java methods are executed.

⇒ structure of a Java program

```
import Java.io.*i → headerfile
class Sum → declare a class
{
Public static void main(string args[])//declared main func.
{
int a,b,c;
System.out.println("welcome to Java"); //Code of the program.
}
}
```

→ In the above structure, first we declare a header file after that, we declare a class with a name sum, in that class we declare a main function, in that main function, we declare sum variables and methods as the code of a program.

→ A main() function in Java can have the following properties:-

*public:- If we declare public, before the main() function it is available anywhere in the Java program. In Java by-default all the classes and methods can be known as public.

*static:- The static methods are executed only ones in a program. The main method of Java executed only one's through out the entire Java program. Hence it must be declared as static.

*void:- Before a main() function, if we declare a void the main() function does not return any value.

*string:- It can be used for representing a group of characters.

*Args[]:-

→ It is used to represents, the command line argument program.

→ By using the above structure, we can declare a Java program for addition of two number, can be represented as-

```
import Java.io.*;
class Sum
{
 Public static void main (Strings args[])
 { int a=10, b=20, c;
   c=a+b;
```

```
        System.out.println("Sum is:"+c);
    }
}
```

⇒ **Variable**

A variable is a name of the memory location that can be used to store some value.

* Rules for creating a variable:-
→ A variable must be begins with a alphabet letter.
→ digits can not be used as variable name's.
→ special symbol are not allowed for declaring a variable, only underscore (_) symbol is allowed while declaring a variable.

* **creating a variable:-**
→ The syntax for creating a variable is-

data type variable name=value;

→ int a=10;
by using the above syntax we declare a variable with an integer data type and we assign a value ten(10) to a variable a.

* **Types of variable:-**
→ A variable can be classfied into four types-
① Global variable
② Local variable
③ Instance variable
④ Static variable.

① **Global variable:-**
If a variable is declare outside of the class that variable is known as Global variable.

eg:
```
int a;
class ab
{
=
}
```

② Local Variable:-

If a variable is declare inside of the class or if a variable is declare within a method, that variable is known as local variable.

eg:
```
class ab
{
int ab;
}
```

③ Instance Variable:-

If a variable is declare inside the class but outside the method that variable is known as instance variable.

eg:
```
class ab
{
int a
void show()
{
=
}
}
```

④ Static Variable:-

If a variable is declare by using the keyboard static that variable is known as static variable.

eg:
```
class ab
{
static int a;
}
```

⇒ Data types in Java.

or primitive data types

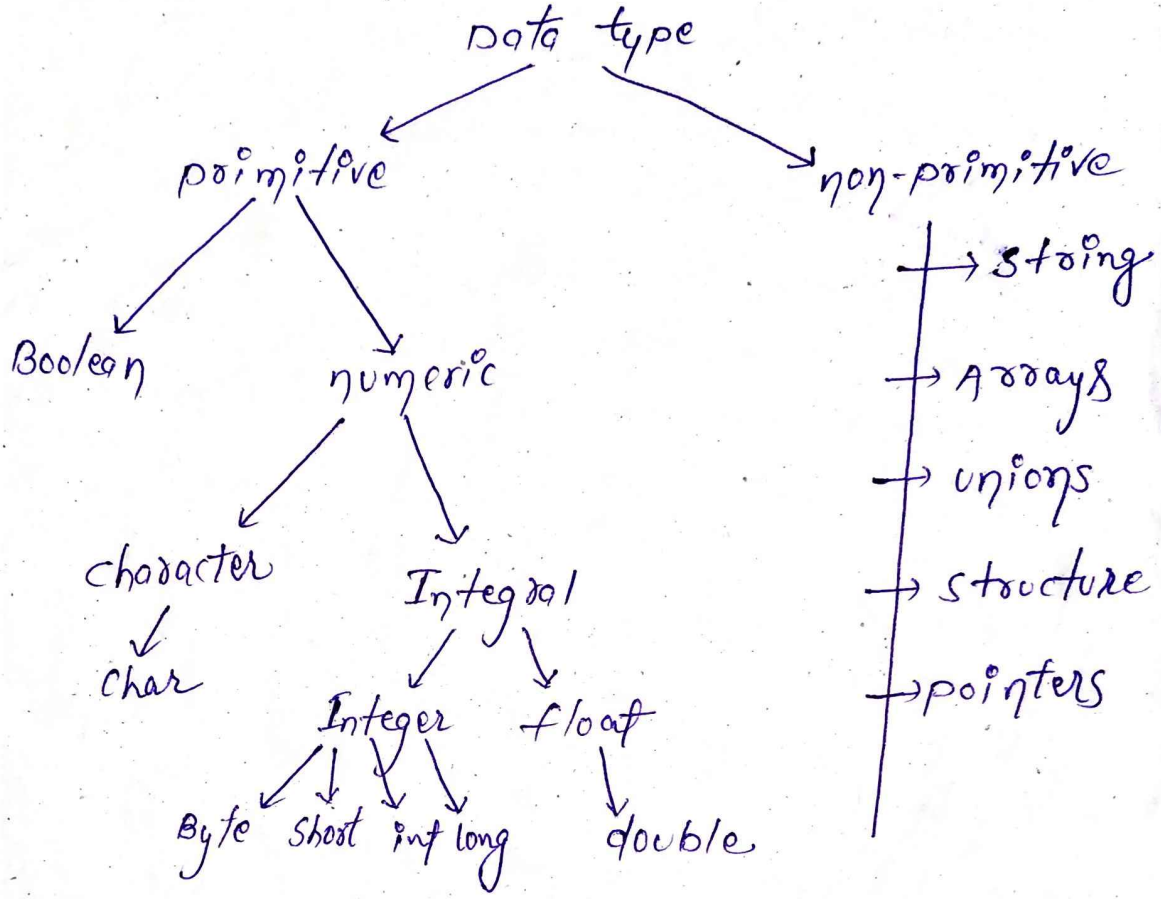


fig:- different type of data type

03/12/18 ⇒ Data type

A data type can be used to represents the type of the value that is stored in a variable.

→ in the above diagram to a represents the total data type that are available in java. Here we are discussed only the primitive d.t that are available in Java.

→ A primitive data can be classified as two types:-

① Boolean data type

A Boolean d.t can represents either true or false value.

② Numeric data type

A numeric data type can be classified into two types:-

(i) character data type (char):- A character data type can be used to store a single character. The size of the char is 1 byte

(ii) Integral :- An integer data type can be classified into four types-

(a) Byte
(b) Sharp byte
(c) int
(d) long - int.

→ The above four d.t can be used to store an integer value. means - a number without decimal point.

→ The size of above d.t are:-

data type	memory size
Byte	1 byte or 8 bits
sharping	2 byte
integer	2 byte
long int	8 byte

⇒ float data type

→ The floating point d.T further classified as double d.T. Both float & double are used to store decimal values means - A number with decimal point.

→ The memory size of float d.T is 4-bytes and The memory size of double D.T is 8-bytes.

→ The difference between a float & double is, A float can represents upto 8-digits after the decimal point. Where as a double can represents 16-digits after the decimal point.

⇒ Identifiers :-

An identifiers refers the name of the variable, methods, packages and interfaces.

eg:

```
class sum
{
  int a, b;
  void show();
}
```

in the above example the variables a & b & the methods show are the identifier.

⇒ Literals :-

* A literal represents a value that is stored in a variable.

* A literal can be classified into 5-types

① Integer literals :- An integer literal represent a value without decimal points.

② float literals :- A float literal represents a value with decimal point.

③ character literals:- A character literal can be used to store a single character.

④ string literals:- A string literal can represents a group of characters.

⑤ Boolean literals:- A Boolean literal can represents ~~can represent~~ either true or false values.

Note:-

In literals an octal number system can be represents by using alphabet letter '0' before the number and the hexa-decimal number system can be represent by using '0x' before the numbers.

⇒ operators

An operator is a symbol that can be used to perform some mathematical calculations.

eg:- A+B

in the above example the symbol '+' is an operator and A&B is known as operands.

→ Types of operator:-

① Arithmatic operator:- An arithmatic operator can be used to perform some arithmatic operation such as- Addition, Substraction, multiplication, division & module-division.

* Some of the arithmatic operations are:-

operator	Name of the op.	example
+	Addition	a+b
-	Substraction	a-b
*	multiplication	a*b
/	division	a/b
%	module divpsion	a%b

4/12/18

⇒ Relational operator

The R.O can be used to compare b/w the two values. Some of the R.O are:-

operator	Name of the operator
<	less than
>	greater than
<=	less than equal
>=	
!=	

⇒ logical operator

logical operators can be used to construct compound condition. A Compound condition is a combination of several simple condition.

eg: if (a=1) && (b=1)

simple condition — simple condition.

operator	Name of O.P
&&	and operator
||	or "
!	not "

⇒ Assignment operator (=)

An A.O can also be known as relational operator which can be used to compare two values and also used to assign values into a variable.

eg: x = 5

⇒ unary minus operator (-)

By using this operator we can convert a positive value into a negative value and a negative value into a positive value.

eg:

```
class SS
{
    Public static void main(string args[])
    {
        int x=5;
        System.out.println(-x)
            "   "   "   (-(-10));
    }
}
```

⇒ Increment operator

An increment operator can be used to increase the value by 1. Incrementing a value can be done in two ways.

① pre-Increment operator

The pre-increment operator can increase the value by 1 by using the pre-increment operator. if you consider an x value is 2 by applying the P.I.O first the value is incremented by 1 & the incremented value is stored into the variable.

② post-increment operator

in post increment operator first the value is assigned to the variable after that it increment the value by one.

eg: class ab

```
{
  Public static void main(string args[])
  {
    int x=2;
     ++x;
    System.out.println("After pre-incremented the
                        value is", ++x);
    x++
    System.out.println("After post-incrementing the
                        value is", +x);
  }
}
```

O/P:

⇒ Decrement operator

This operator is used to decrement the values of a variable by 1. for example if we take x=2 whe we use pre-decrement operator ie the value becomes 1, then if we are use post-decrement operator then the value becomes 0.

Note:

$x++ = x = x+1$

$x-- = x = x-1$

eg: class ab

```
{
  Public static void main(string args[])
  {
    int x=2;
```

```
        --x;
        System.out.println("After pre-decrementing
                the value is", -x);
        x--;
        System.out.println("After post-decrementing
                the value is", -x);
    }
}
```

O/P:

⇒ Ternary or conditional operator (?, :)

A ternary or conditional operator can be defined with the help of ? & colon (:). The syntax of this operator is —

```
expression ? statement1 : statement2;
```

in the above syntax, if the expression is true the first statement will be executed, otherwise the second statement will be executed.

05/12/08

⇒ precedence & Associativity rules :—

The priority of an operator can be given the levels of the operator. The operators *, /, % will come under the level 1 and the operators +, - will come under level 2.

→ If the same levels of operators (os) in equation then it is choice of the user to give the priority of those operator.

→ If an equation having the parenthesis and the braces then we have to give the priority for parenthesis and braces.

eg: step1 : 2-5*4/5

step2 : 2-20/5

step3: 2-4

step4: -2

in the above step the equation contains minus, *, / operators. The *, / operators are belongs to the same level (level1) and the both operators have same priority then it the choiche of the user to give the priority among to operators. Here I am giving the priority for '*' operators, then the above equation can be represented assure in step2.

→ In step2 the equation contains '-' & '/' operator then we are giving the priority to / division as it belongs to level1. Then the equation can be represented in step3.

→ In step3 the equation contains only one operator that is '-' operator then there is no choice to consider the priority of the operator as the equation contains only one operator then if we solve the equation according to the operator the result of the above equation can be shown.

⇒ primitive Type conversion and casting

Converting one data type into another data type is known as Type conversion (or) Type casting.

→ casting of primitive data type

We can convert one primitive data type into another primitive data type, This can be done in two ways.

① **widing:-**

converting a lower data type into a higher data type is known as widing. This can be represented as —

```
char ch='A';
    int num=(int)ch;
```

in the above example we convert a lower data type char into a higher data type int, for this purpose we use ~~char~~ cast operator by writing ~~operator~~ in the simple braces before the variable ch. Now the ch is converted into 'int' type an assign the converted value to the variable num.

② **Narrowing:-**

Converting a higher data type into a lower data type is known as Narrowing. This can be represented as —

```
int num=65;
    char ch=(char)num;
```

in the above example we are converting a higher data type int into a lower data type char. for this purpose we use cast operator char by writing it in simple braces before the variable 'num'. Now the 'num' is converted into char type and assign converted value to the variable ch

⇒ Expression

An expression is a combination of variables, literals and operators.

eg: int x=2;
int z=x*y;

in the above example 'x' is a variable and 2 is the value that stored into a variable can be known as literals and z=x*y is known as an expression as it contains variables. and operators

06/12/18 ⇒ Flow Control :-

* if-else: The if-else statement can be used to check the condition. The syntax of if-else statement is :-

```
if (condition)
statement 1;
else
statement 2;
```

in the above syntax first the if-statement will check the condition, if the condition is True the Statement 1 will be executed otherwise Statement-2 will be executed

program eg:-
```
class demo
{
  Public static void main (string args[])
  {
    int num=-5;
     if (num>=0)
     System.out.println("num is positive");
     else
     System.out.println("num is negative");
  }
}
```
o/p:- num is negative.

06/12/18

* while :-

→ The while statement is used to repeate a group of statement as long as the condition is true.

→ If the condition is false then it exit from the loop.

Syntax:-
```
while(condition)
{
  statement (s);
}
```

eg:-
```
class demo1
{
  Public static void main (string args[])
  {
    int n=0;
    while (n<10)
    {
     x++;
    System.out.println(n);
  }
  }
}
```
o/p:- 1 2 3 4 5 6 7 8 9 10

* do-while :-

In do-while statement first the statement will be printed after printing the statement. The condition will be checked. by using the do-while statement the programmer does not have the control on the program.

Syntax :-

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

program eg:-

```
class demo1
{
 public static void main(string args[])
 {
  int x=0;
  do
  {
   x++;
   System.out.println(x);
  }while(x<10);
 }
}
```

o/p :-

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

* for loop :-

The for loop execute a block of statements untill the condition is true. it contains three parts initallisation, condition and increment and decrement.

Syntax:-

```
for(initilization; condition; increament/decrement)
{
  statements;
}
```

```
for (int i=10; i<10; i++/i--)
{
  =
  =
}
```

by using the above syntax we can represents a for loop as:-

program:-

eg:-

```
class ab
{
  Public static void main (string args[])
  {
    for(int x=0; x<10; x++)
    {
      system.out.println(x);
    }
  }
}
```

o/p:-

0
1
2
3
4
5
6
7
8
9

statement

* Switch:-

When they are several options we have to choose one options from the available one, we can use the switch statement depending upon the selected option a particular task can be perfo

Syntax:

```
Switch(variable)
{
 case1:
  =
 case2:
  =
 case3:
  =
 default:
   default statement;
}
```

program eg:

```
class ab
{
 Public static void main(string args[])
 {
  char color='g';
  swith (color)
  {
  case 'B':
  System.out.println("Blue")
  case'g':
  System.out.println("green");
  case 'R':
  System.out.println("Red");
  case 'P':
  System.out.println("pink");
  default:
  System.out.println("No Color");
 }
}}
```

o/p:- green
red
pink
no color

* Break statement:-

The Break statement can be used to come out of the loop (or) case that are written in a switch statement.

program:

eg:

```
class ab
{
Public static Void main (string args[])
{
 char color='g';
 Switch (color)
 {
 case 'B':
 System.out.println("Blue");
 Break;
 case 'g':
 System.out.println("green");
 Break;
 case 'R':
 System.out.println("Red");
 Break;
 case 'P':
 System.out.println("pink");
 Break;
 default:
  System.out.println("No Color");
 }
}
}
```

op:- green