**JAVA**

* Java is high level programing language.
* Java is Object oriented programming language.
* Java was developed by “James Gosling” in the 1995 and his team is an organization has called “**Sun microsystem**” with project called “**Green**”.

Example1: class Cricket

{

public static void main(String a[])

{

System.out.println(“Indian has best cricket team. ”);

}

}

Output: Indian has best cricket team.

Example2:

class Map

{

public static void main(String a[])

{

System.out.println(“Belagavi to Bengalore. ”);

}

}

Output:

Belagavi to Bengalore.

**PROGRAM**

What is mean by program?

* A chain of steps or set of activities combine all together in order to complete or achieve specific task.
* Program will be always **static** state.
* Program in execution is known as **processor**.
* Processor **runs** the **process.**

**JAVA PLATFORM:**

In the java there are three types of platforms.

Platform

JSE JME

JEE

* JSE- Java Standard Edition.

Ex:Desktop Application

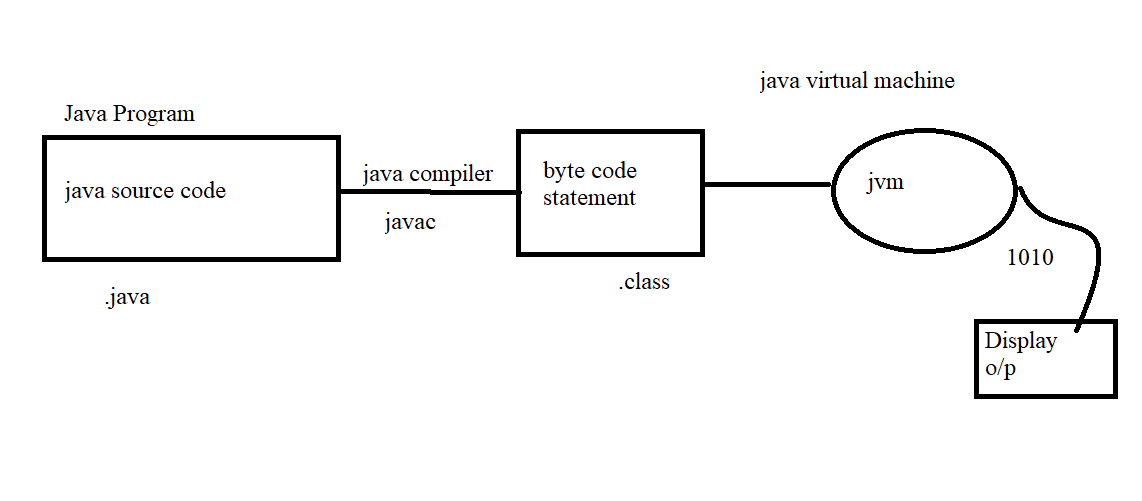
* JEE-Java Enterprises Edition.

Ex:Web application and Enterprises.

* JME-Java Micro Edition.

Ex :Utility software. (Android Apllications)

Working of program



* **Object** oriented programming language is creating objects that contain both data and methods.
* Object is real world entity.
* Object is a related with real world.

**Program Language:**

* Provide Syntax and semantics to write program.

**Object Oriented :**

* It is an approach which is used to design a software or program.

**Variable:**

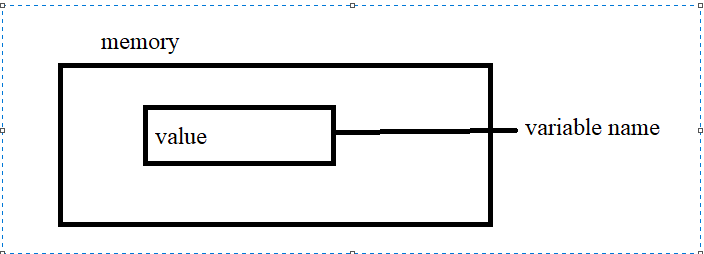
* It is one which holds the value or container which can be used in the letter stage for the program language.

Syntax: **Datatype variableName=value;**

There are two steps of variables

1. **Declare the variable.**
2. **Initialize variable.**

It is also considered as Register area in the memory location.



There are four types of variables

1. local variables
2. static variables
3. instance variables
4. parameter variables

example:

class PrimeMinister

{

static String name=”Narendra Modi”

public static void main(String a[])

{

int age=70;

System.out.println(name);

System.out.println(age);

}

}

**DataTypes**

* Data types it is used to represent the type of data to be stored in the memory.

There are two types of data types:

|  |  |  |
| --- | --- | --- |
| **Primitive DT** |  | **Non Primitive DT** |
| byte- 1byte \_ | -128 to 127 | String |
| short- 2 bytes 16bit |  | Interface type |
| int- 4 bytes 32 bit |  | Class type |
| long- 8 bytes 64 bit |  | Arrays etc |
| float -4 bytes 32bit | stores 6 to 7 decimal numbers |  |
| double -8 bytes 64bit | Stores 15 decimal no |  |
| char -2bytes 16 bit | Stores a single character/letter or ASCII values |  |
| boolean -1bit 8 bit | Stores true / false |  |

Example: class Appu{

public static void main(String a[])

{

String name=”Appayya”;

short age=24; //int

float weight=56.76f;

double height=173.2;

long phonNo=8217237037L;

boolean isAlive=true;

System.out.println(“My name is “ + name);

System.out.println(“My age “ + age);

System.out.println(“My height is “ +height);

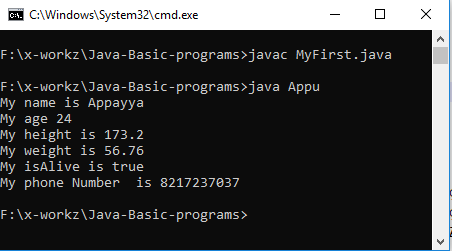
System.out.println(“My weight is “ +weight);

System.out.println(“My phone Number is “ + phoneNo);

}

}

Output:



* All the non-primitive data types are default values are **null.**

**Char:** It is enclosed with single quote**(‘’).**

* It is used uppercase and lowercase later**(A-Z, a-z).**
* It is used Numbers**(0-9)**.
* Default value char is space **( )**
* It is used special characters ($ )

**Rules of Local variables**

In side the main () method you can declare the variables as well as initialize the variable is known as local variable.

Example: int i=10;

class Cell

{

Static String cellName=”MI”;

Public static void main(String a[])

{

int price=25000;

System.out.println(“Phone Name:”+cellName);

System.out.println(“Phone Price: ”+price);

}

}

Output:

Phone Name: MI

Phone Price: 25000

**Method:**

Method are block of statements or collections of statements that are grouped together to perform some functionalities.

Syntax:<access-modifiers> return type methodname(p1,p2,p3,,,,,,pn)

{

//block of statements

return value;

}

Example

class AdditionOfTwoNumbers

{

public static void main(String a[])

{

add(9,8);

add(5,6);

}

public static add(int a, int b)

**{**

int c=a+b;

System.out.println(c);

}

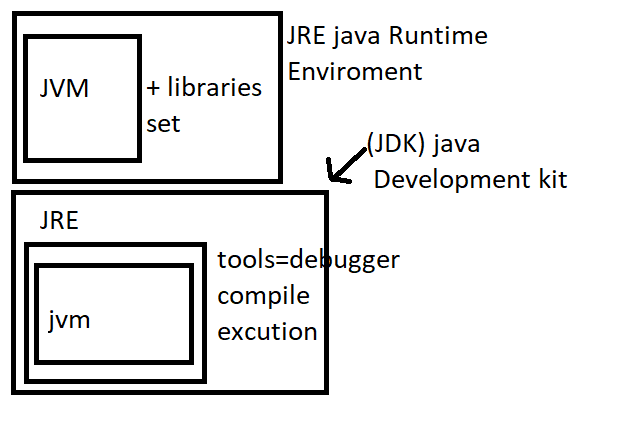
* It helps us to reuse the code.

Rules to write a **method**

* We can write a method inside method
* We can call a method inside another method multiple number of times.
* Method should be written inside a class.

**Dtae :30/01/2021**

Q. Difference between JRE JDK JVM



* JVM- It is run environment and it is used to run the help of libraries set.
* Whenever the code want to debugger or modify JDK is needed. without JDK we can’t modify.
* When the code will be compile, the compiler will be generated **bytecode** or **.class** file
* Jvm – will be converted bytecode to machine code.
* JRE will be run byte code.

**JVM: [**Java Virtual Machine**]**

JVM is provide runtime environment in which java bytecode can be executed.

Task of JVM

* Loads the code.
* Verify the code.
* Execute the code.
* Provide the run time environment.
* JVM is platform defendant.

**JRE:** [Java Runtime Environment]

* JRE=jvm+librariesset to run any java program or java code JRE is minimum required.
* JRE contains a set of libraries at JVM uses at the runtime.
* JRE is physical exists.
* JRE is platform defendant.

**JDK:** [Java Development Kit]

* It contains JDK=JRE + Development kit**.**
* JRE=JVM + libraries
* Development Tools
* Debugger
* Compile
* Execution etc.
* JDK is physical exists
* It is full featured development

**Java Fundamentals: or Tokens**

* Java fundamentals are required to develop or write a java program.
* Identifiers
* Keywords
* Literals
* Variables
* Data types
* Methods
* Oops [inheritance, polymorphism encapsulation and abstraction]
* Operator
* Comments
* Separators

**Identifiers**

It is a name given to class name, variable name , method name package name etc.

Rule to define identifiers

A-Z , a-z , 0-9 , ( \_underscore )($ doller)

If we are using other character we will get compile time error

Example :

class Speaker\_$-----------------valid

class Speaker##---------------------- un valid

identifiers are not allowed to starts with digits

example:

class 123Speaker------------invalid

class Speaker\_32230-------valid

identifiers are case-sensitive as java program language treated as case-sensitive.

No duplicate value in java

**Method Overloading**

Methods with same name having different number of parameters is called as method overloading.

It is used increase readability.

Method overloading it is used to achieve “compile time polymorphism”.

**Different ways of method overloading**

Can be achieved by verifying parameters list one of the below.

**Number of Parameter**

Data types of parameters

The sequence of data type of parameter

Example :

public static void ring(String a[])

public static void ring(String name)

it is doesn’t matter to overloading whatever.

Literals – java are a sequence of characters like digits, letters, and other characters.

A literal are the constant value assigned to the constant variables.

Literal are fixed values that cannot modified.

**Reserve words**

**Keywords -50 reserve literals -3**

Used keywords -48unused keywords-2 True Null False

Const

Goto

Keywords are the predefined words or reserve words having any specific meaning are functionality.

**Literals** The value assigning to variable is known as literals.

-Integral literal

- Character literal

- String literal

-Floating literal

- Boolean literal

Ex: Flights

**Arrays:**

It is a group of similar type of data.

Syntax :

Datatype variable [] ={size};

Datatype []variable ={size};

Datatype[] variable ={size};

Example:

class Flights

{

static String name[]={“Deccan”,”AirAsia”,”SpaceJet”};

public static void main(String []a)

{

System.out.println(name[0]);

}

Arrays has property called as **length** which return size of array.

Array elements can be accessed using index.

Index value will always starts ‘0’.