Topic: Variables, Data Types, method

Date: 04/04/2023

1.Variables:

Variables are nothing but piece of memory use to store information.

one variable can store 1 information at a time.

Variables also used in information reusability.

To utilise variables in java programming language we need to follow below steps:

1. Variable declaration (Allocating/Reserving memory)
2. Variable Initialization (Assigning or Inserting value)
3. Variable Usage

Note:-

According to all programming language dealing with information directly is not a good practice to overcome this variable are introduced.

Data Types:

Datatype are used to represent type of data or information which we are going to use in our java program

In java programming it is mandatory to declare datatype before declaration of variable.

In java datatypes are classified into two types :

1. Primitive datatype.
2. Non-primitive datatype.

1.Primitive datatype:

There are 8 type of primitive datatypes(byte,short,int,long,float,double,char,boolean).

All the primitive datatypes are keywords.

* Memory size of primitive datatype are fix.

The types of primitive datatype are:

Note:- keyword starts with lower case

Primitive datatype starts with lower case

syntax: datatype variablename;

1.(Numeric + Non-decimal):-

Ex: 80,85,10,5..etc

	Data Type	Size	
1.	byte	1 byte	
2.	short	2 bytes	
3.	int(imp)	4 bytes	
4.	long	8 bytes	l

1GB=1024MB

1MB=1024KB

1KB=1024Byte

1Byte= 8bit

2.(Numeric + decimal):-

Ex: 22.5,22.8,6.4....

5.	float(imp)	4 byte	f
6.	double	8 byte	d

3. Single Character :-

Ex: A,B,X,Z.

7. char 2 byte

4. Conditional:-

Ex: true,false.

8. boolean 1 bit

----

2. Non-primitive datatype:

There are 2 types of non primitive datatypes .

all the Non primitive datatypes are identifiers.

* Memory size of non primitive datatype is not defined or not fix.

Note: Identifier starts with capital letter.

Non-primitive datatype starts with capital letter.

e.g. String, className

```
package Variable;

public class sample1
{
     public static void main(String[] args)
     {

          //variable declaration
             String sname;                //dataType
variableName;
             int  srollNum;
             float sper;
             char sgrade;

          //variable initialization
             sname="rahul";                   //variableName =
"rahul";
             srollNum=102;
             sper=65.5f;
             sgrade='A';

            //variable usage
             System.out.println("Student name="+sname);
             System.out.println("Student roll num="+srollNum);
             System.out.println("Student
percentage="+sper+"%");
             System.out.println("Student grade="+sgrade);
             System.out.println("Student grade="+sgrade);
//reuse
     }
}
```

```
package Variable;

public class sample2
{
     public static void main(String[] args)
     {
          //variable declaration & initialization in single
step
             String sname="rahul";               //dataType
variableName;
             int  srollNum=101;
             float sper=65.5f;
             char sgrade='A';

            //variable usage
             System.out.println("Student name="+sname);
             System.out.println("Student roll num="+srollNum);
             System.out.println("Student
percentage="+sper+"%");
             System.out.println("Student grade="+sgrade);
             System.out.println("Student grade="+sgrade);
//reuse
     }
}
```

Methods:

A method is a block of code which only runs when it is called.

Methods are used to perform certain actions, and they are also known as functions.

You can pass data, known as parameters, into a method.

Why use methods? To reuse code: define the code once, and use it many times.

1. **main method** (pre-defined)
   In any Java program, the main() method is the starting point from where compiler starts program execution.
   So, the compiler needs to call the main() method.

   without main method we can't run any java program.

2. **Regular method** (user defined)

   1. static regular method
      1. static regular method call from same class --> methodname();
      2. static method call from diffrent/another class -->className.methodname();
   2. non- static regular method
      3. non-static method call from same class --> 1. create object of same class 2. objectname.methodname();
      4. non-static method call from diffrent/another class --> 1. create object of diff class 2. diffClassObjectName.methodname();

Note: At the time of program execution main method is going to get executed automatically where as regular methods are not going to get executed automatically.

At the time of program execution priority is scheduled for main method only.

To call a regular method we need to make call method call from main method,

until unless if the method call is not made regular method will not get executed.

regular methods can be called multiple times.

5. method without/zero parameter
6. method with parameter.
7. method with return type

```
package Methods;

public class sample1
{
     //main method
     public static void main(String[] args)   //pre-defined
method  //method declaration
     {
             System.out.println("hi"); //method body

             //1: static regular method call from same class
             m1();      // methodname();

            //2: static regular method call from diff class
             sample2.m2();     //diffclassname.methodname()

             //3: non-static regular method call from same
class
              //classname objectname=new classname();
//syntax of object creation
             sample1 s1=new sample1();
             s1.m3();

             sample2 s2=new sample2();
             s2.m4();

             System.out.println("Hello");
     }


     //static ->regular method
     public static void m1()   //user defined
     {
          System.out.println("running static regular method m1
from same class");
     }


     //non-static ->regular method
     public void m3()   //user defined
     {
          System.out.println("running non-static regular
method m3 from same class");
     }
}
```

```
package Methods;

public class sample2
{

    //static ->regular method
    public static void m2()   //user defined
    {
        System.out.println("running static regular method m2
from diff class");
    }


    //non-static ->regular method
    public void m4()   //user defined
    {
        System.out.println("running non-static regular
method m4 from diff class");
    }

}
```

```
package Methods;

public class sample3
{
     //method with return Type

     public static void main(String[] args)
     {
          mul(10,20);     //initialization
          mul(5,6);

          System.out.println("-------");

          sample3 s3=new sample3();
          s3.sub(10, 8);
          s3.sub(9, 15);

     }




     //method with 2 int (int , int) parameter
     public static void mul(int num1, int num2)    //variable
declaration
     {
          System.out.println(num1*num2);
     }


     //method with 2 int (int , int) parameter
     public void sub(int num1, int num2)    //variable
declaration
     {
          System.out.println(num1-num2);
     }
}
```

Control statement:

```java
package ConstrolStatement;

public class example1_if
{
	public static void main(String[] args)
	{

		int marks=20;

		//  20>=35
		if(marks>=35)
		{
			System.out.println("pass");
		}
	}
}
```

```java
package ConstrolStatement;

public class example2_if_else
{
	public static void main(String[] args)
	{

		int marks=40;

		// 40>=35
		if(marks>=35)
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}
}
```