**Data types**

The data type is useful for storing the data and specifying the size and type of data.

**There are two types of data available in Java:**

* **Primitive**
* **Non-primitive/object type.**

In Java, all the variables should be declared before using them.

**For Example,** **int a=1**; Here, we are insisting compiler that “a” exist and it is initialed with value 1. Based on this datatype (**int**), the compiler will allocate memory.

**Primitive Data Types**

* **Char:** It is a single 16-bit Unicode character.
* **Boolean:** It is one bit and this can have true and false values.
* **Byte:** Byte is 8-bit two’s complement integer. This is mainly useful in large arrays to save memory and its range is from -128 to 128.
* **Short:** This type is also more useful in saving memory in large arrays and it is 16-bit two’s complement signed integer.
* **Int**: 32-bit signed two’s complement.
* **Long:** 64-bit two’s complement.

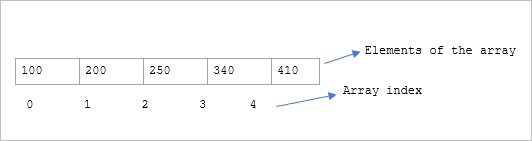
**Floating point data types**

* **float and double:**Float is 32-bit single precision floating point. Double is 64-bit double precision floating point number.
* **Data Types Range**

| **DATA TYPE** | **DEFUALT VALUE** | **SIZE** | **EXAMPLE LITERALS** | **RANGE** |
| --- | --- | --- | --- | --- |
| boolean |  | 1 bit | true, false | - |
| byte | 0 | 8 bits | (none) | -128 to 127 |
| char | \u0000 | 16 bits | 'a', '\u0041', '\101', '\\', '\'', '\n', 'ß' | '\u0000' -'\uffff' |
| short | 0 | 16 bits | (none) | -32,768 to 32,767 |
| int | 0 | 32 bits | -2, -1, 0, 1, 2 | -2,147,483,648 to 2,147,483,647 |
| long | 0L | 64 bits | -2L, -1L, 0L, 1L, 2L | -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 |
| float | 0.0f | 32 bits | 1.23e100f, -1.23e-100f, .3f, 3.14F | 3.40282347 x 10 38 , 1.40239846 x 10 -45f |
| double | 0.0d | 64 bits | 1.23456e300d, -1.23456e-300d, 1e1d | 1.7976931348623157 x 10 308 , 4.9406564584124654 x 10 -324 |

* **Example Program for Datatypes:**

|  |
| --- |
| public class DemoDataType {    public static void main(String[] args) {    // Auto-generated method stub  char C='M';  char a=65535;  byte b1=-128;  byte b2=127;  int i1=-2147483648;  int i2=2147483647;  short s1=-32768;  short s2=32767;  long  L2=9223372036854775807L;  long L1=-9223372036854775808L;  float f1=3.40282346638528860e+38f;  //float f3=3.40282347e+38f;  float f2=1.40129846432481707e-45f;  double d=4.37346473;    System.out.println("char is: " + C);  System.out.println("int low range is: " + i1);  System.out.println("int high ramge is: " + i2);  System.out.println("byte is: " + b1);  System.out.println("short low range: " + s1);  System.out.println("short high range: " + s2);  System.out.println("Long high range is: " + L2);  System.out.println("Long Low range is: " + L1);  System.out.println("double is: " + d);  }  } |

* **Arrays**
* It is an object that can hold a fixed number of values of the similar data type. The array can be created by a new keyword.
* Int array = new int[5];
* Here the type of array is int and the length is 5.
* 
* The length of the array should be initialized while an array is created.

|  |
| --- |
| public class ArrayDemo {    public static void main(String[] args) {    // TODO Auto-generated method stub  int Array[] = new int[5];  Array[0]=100;  Array[1]=200;  Array[2 ]=300;  Array[3]=400;  Array[4]=500;    for(int i=0;i<Array.length;i++)  {  System.out.println(Array[i]);  }    //also you can declare in this way  int another\_Array[]= {100,200,300,400,500};    for(int i=0;i< another\_Array.length;i++)  {  System.out.println(Array[i]);  }  }  } |

* **Loops**
* Loops are generally used to execute the same set of instructions for many times.
* **For Example,** if you want to print “software testing help blog” sentence for five times, you need to specify like this.
* **Without Loops:**

|  |
| --- |
| public static void main(String[] args) {    System.out.println(“software testing help blog”);  System.out.println(“software testing help blog”);  System.out.println(“software testing help blog”);  System.out.println(“software testing help blog”);  System.out.println(“software testing help blog”);  } |

* **With Loops:**

|  |
| --- |
| public static void main (String [] args) {    for (int i=0; i<5; i++) {  System.out.println(“software testing help blog”);  }  } |

* For loop is useful for iterating values.
* **For Loop:**

|  |
| --- |
| for (initialization; termination;  increment) {  statement(s)} |

* **While Loop:**
* While Loop checks the Boolean condition first and if that condition is true, the statements will be executed repeatedly until the condition fails.

|  |
| --- |
| While (condition) {  Statement to be executed  } |

* **Do While Loop:**
* In the [do while loop](https://www.softwaretestinghelp.com/java-do-while-loop/), the statement will be executed first and then the condition will be checked. If the condition is true, the statement will be executed again. Otherwise, execution will come out of the loop.

|  |
| --- |
| do {  Statements to be executed  }  while(condition); |

* **Loops Demo Program:**

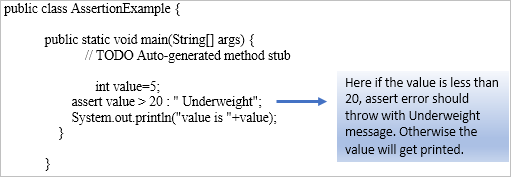
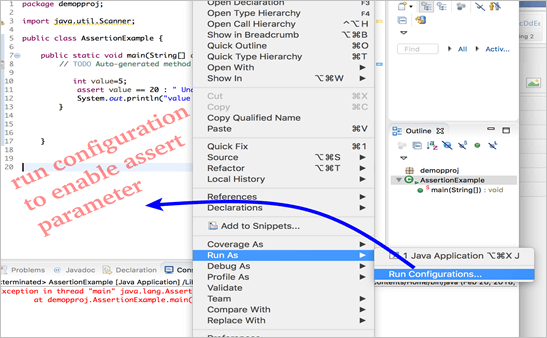
|  |
| --- |
| package demopproj;  public class LoopsDemo {    public static void main(String args[]) {    //for loop demo  for(int i=0;i<5;i++) {  System.out.println("iteration  number : " + i);  }    //while loop demo  int count\_while = 1;  while (count\_while < 3) {  System.out.println("count\_while is: " + count\_while);  count\_while++;  }    //do while demo  int count\_dowhile = 1;  do {  System.out.println("count\_dowhile is: " + count\_dowhile);  count\_dowhile++;  } while (count\_dowhile < 3);    //for each demo  int[] numbers =  {1,2,3,4,5};  for (int item : numbers) {  System.out.println("item number is: " + item);  }  }  } |

* **Switch Statement**
* The switch statement will have the number of execution. Based on the value input, the switch case will be executed.
* Here the value is 8, the case 8 will be executed.

|  |
| --- |
| public class SwitchDemo {    public static void main(String[] args) {    //switch demo  int item = 8;  String product;  switch (item) {    case 1:  product = "iphone5";  break;  case 2:  product = "iphone6";  break;  case 3:  product = "iphone7";  break;  case 4:  product = "iphone8";  break;  case 5:  product = "samsung7";  break;  case 6:  product = "samsung8";  break;  case 7:  product = "nokia";  break;  case 8:  product = "macbook";  break;  default: product = "motoG";  break;  }    System.out.println(product);  } |

* **Assertion in Java**
* An assertion is used to test the assumption that we made while programming.
* **For Example,** we receive input from third-party. We assume that input the won’t be null.
* To test that Program, assertion is used.
* **Assertion method:**1 assert expression;
* **Assertion Method:** 2 assert expression1: expression2;

|  |
| --- |
| public class AssertionExample {  public static void main(String[] args) {  // TODO Auto-generated method stub    int value=5;  assert value > 20 : "Underweight";  System.out.println("value is"+value);  }  } |

* 
* To test this program, the assertion should be enabled. To enable assertion, a parameter should be added in the run configuration.
* 
* 