Java

19/10/2018

**Any queries/questions from the previous session ?**

>> Goal of today’s session – Introduce you to the basics of Java programming

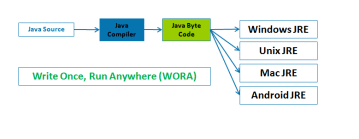
**>> Q? Have all of you installed JDK 8 on your respective machines ? We will install the JDK on those machines that don’t have it. JAVA\_HOME, Path needs to be set**

**On Mac – what are the steps to be followed ?**

>> What is the JDK ?

>> Compile and run your Java programs

>> Write once run anywhere

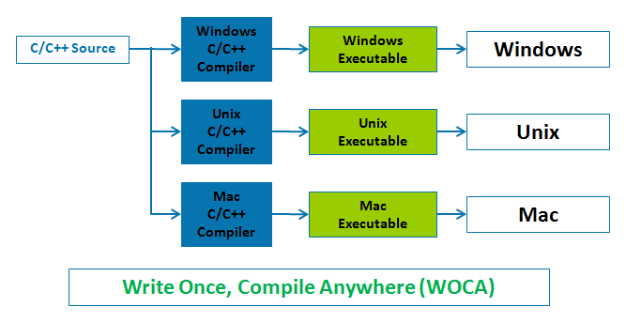


**java bytecode** is the machine **code** in the form of a .class file. The JVM executes this bytecode.

**>> Q ? The same java file is compiled on Mac and Windows . Will the .class file be the same ?**

**Let us create a simple program to demonstrate the above concepts**

|  |
| --- |
| 1. **Public class** ReallySimple{ 2. **public** **static** **void** **main**(String args[]){ 3. System.out.println("Really simple java program"); 4. } 5. } |



Source of the two images - <https://core2advance.wordpress.com/2016/03/14/creation-of-java-for-platform-independence-wora/>

*FYI – Java was created for platform independence and not web development, data sciences, etc.*

**Q? Is Java a popular language ?**

Yes.

For the students, as part of your assignment, please find out how Java is being used ? . It is widely used across the industry and for developing different kinds of business applications and IT products.

**Q? Do I have to code everything from scratch or are there libraries that I can use in my programs ?**

No. Java comes with a rich standard library that is being continuously enhanced that developers can refer to.

For the students, as part of your assignment, each of you find out about 1 library/module ? .

>> We can create Java code using a number of different editors

1. Notepad
2. Notepad++
3. Eclipse IDE [Will be covered later on in the course]
4. NetBeans IDE
5. IntelliJ

We will start with Notepad(++) as it is easier for beginners. Later on, we will move to Eclipse. 2-3 sessions will be conducted using Notepad.

Using IDEs for applications improves developer productivity and make it easy to compile, test and run programs.

>> Creating applications in Java

>> Applications are simply a collection of programs

>> We’ll start with really simple applications which consist of a program. Later on, the apps will have more than 1 program

**Q? In general, what do you think a program consist of ?**

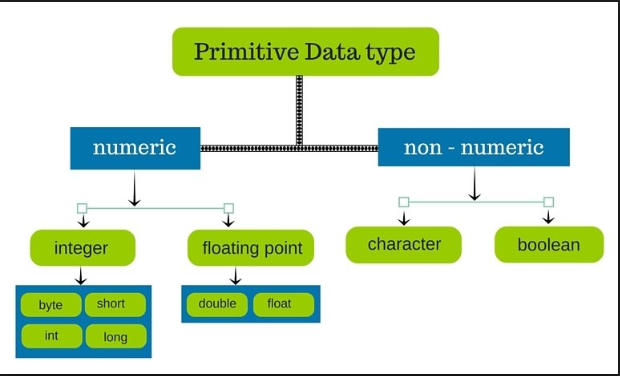
>> Data and actions performed upon that data

>> Students to provide some examples

>> To handle data, Java provides two data types

>> Primitive Data types

>> Reference Data types



**Exercise – Please find out more about integer data types and floating points. Why do we have so many different primitive data types**

NOTE : Java is an object oriented language. For reasons of performance, it has primitive data types. These are the simplest data types that one can create/use in Java. ***They do not form part of the Class hierarchy in Java [will be explained later in the OOP sessions]***

These primitive data types have no special capabilities and serve just one purpose. Store data for quick retrieval.

We will for the time being use two reference data types – “String” and “Arrays”. These are actually objects that we will use in our programs.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| -1 | -2 | 100 | 0 | 5 | 7 | 7890 | 34 | 23 | -22 |

---------------------------------------------- 19th end --------------------------------------------------------------------------------

JAVA PROGRAMS

1. Write a program which calculates the average of three numbers

>> Is the average an Integer ? NO

>> Can we store the average as an Integer ? YES – Will be covered later

>> Assuming simple logic, would this work for say 10 numbers, 5 numbers - Array with Loops -

1. Given Principal, Rate of Interest and Time. Find the simple interest

>> At run time, can we change the values – YES

>> STRING to INT conversion – TBD

>> Storing the simple interest in an Integer data type (Automatic conversion or “Implicit Conversion” )

1. Any suggestions from the team (let’s try to ensure that all primitive data types are used along with Strings and Arrays]

>> Saving these files on GITHUB

**22nd October 2018**

WEEKEND ASSIGNMENT

## Given two numbers, add, multiple, subtract these two numbers

## If calling prefix = “+91”, then calling country is India

## For a student, if % > 60, then grade = “Ist Class”, else “2nd Class”

## Given a set of numbers, find the highest, 2nd highest number

## Given a set of numbers, list the numbers in ascending or descending order

**Q. Did anybody find out more about numeric data types ? Why do we have so many of them ?**

|  |  |  |  |
| --- | --- | --- | --- |
| **Numeric** | **Byte** |  |  |
|  | **Short** |  |  |
|  | **Int** |  |  |
|  | **Long** |  |  |
|  | **Double** |  |  |
|  | **Float** |  |  |

**23rd – Cover numeric data types in case the team has not understood why there are so many types**

// From the previous session on the 19th

int P = **Integer**.parseInt(args[0]) ;

int R = Integer.parseInt(args[1]) ;

int T = Integer.parseInt(args[2]) ;

// Scanner class – Cover on the 23rd

>> Strings and Arrays

The quick brown fox jumped over the lazy dog

Substring (str1, 40)

Substring (str1,40,3)

>> IF ELSE

>> WHILE

>> DO WHILE

>> FOR

(simple for loops, more complex for loops)

>> SWITCH STATEMENTS

|  |
| --- |
| switch (day) {          case 1:              dayString = "Monday";              break;          case 2:              dayString = "Tuesday";              break;          case 3:              dayString = "Wednesday";              break;          case 4:              dayString = "Thursday";              break;          case 5: |

Assignment – Given the month number, print the month as a String

>> Operators

* Arithmetic Operators
  + Module , ++, --
* Relational Operators
  + ==
* Bitwise Operators (will not be covered)
* Assignment Operators
  + a += b is equivalent to a = a + b.
  + a -= b is equivalent to a = a – b
  + a \*= b is equivalent to a = a \* b
  + a /= b is equivalent to a = a / b
  + a %= b is equivalent to a = a % b
* Logical Operators
  + &&, || and !

SOME JAVA PROGRAMS

|  |
| --- |
| Int x, y = 1 ;  X = ++y ; // x = y++ |

>> More on Arrays and Strings

>> Strings

**Strings**, which are widely used in **Java** programming, are a sequence of characters. In **Java** programming language, **strings** are treated as objects. The **Java** platform provides the **String** class to create and manipulate **strings**.

As String is a class, it comes with a rich set of functions

<https://docs.oracle.com/javase/8/docs/api/java/lang/String.html>

Why are we visiting this link ?. Would like to share a technique that is useful for understanding what a class has to offer

This is the official specification of the String class. We’ll just walk through the amazing set of functions that it provides to manipulate strings.

// The difference between the two will be explained later

String s = “Training” ;

String s = new String (“Training”);

*We will spend time on Strings given the importance of this data type/class in Java*

JAVA PROGRAMS RELATED TO STRINGS

>> Write a program which compares two strings

>> Write a program to retrieve a portion of a string

>> Converts a string to upper case

**----------------------------------- 22nd October ---------------------------------------**

**23rd October 2018**

**>> A step back – Github – Excellent repository for storing code – Has anybody checked it out ?**

>> What did we cover on the 22nd ?

>> Is there a function to reverse a string ?

>> Checkpoint – Any queries on the STRING class ?

>> Assignments that were given on the 22nd

>> Moving to Eclipse as the default IDE –

>> Assignment – Given the month number, print the month as a String

|  |  |  |  |
| --- | --- | --- | --- |
| **Numeric** | **Byte** | **1 byte  -128 to 127** |  |
|  | **Short** | **2 bytes  -32,768 to 32,767** |  |
|  | **Int** | **4 bytes -2,147,483,648 to 2,147,483,647** |  |
|  | **Long** | **8 bytes -9,223,372,036,854,775,808 to 9,223,372,036,854,775,80** |  |
|  | **Double** |  |  |
|  | **Float** |  |  |

>> The For Loop – Let us dive into this topic

|  |
| --- |
| //for loop – This is a really simple example  for(int i=1;i<=10;i++){  System.out.println(i);  } |

|  |
| --- |
| // What does this do ? Goes into an infinite loop. Rarely ever used  for(;;){  //code to be executed  } |

|  |
| --- |
| 1. **public** **class** ForEachExample { 2. **public** **static** **void** main(String[] args) { 3. //Declaring an array 4. **int** arr[]={12,23,44,56,78}; 5. //Printing array using for-each loop 6. **for**(**int** i:arr){ 7. System.out.println(i); 8. } 9. } 10. }   String[] regEX ;    String delta = "xb@cd@ef@pg@testing@training" ;  regEX = delta.split ("@");    for (String strTemp : regEX){  System.out.println(strTemp);  } |

|  |
| --- |
| public class ReverseFor {  public static void main(String args[]){  for (int i= 10 ; i>= 1 ; i--) {  // Have used format instead of println  System.out.format ("\nvalue of i is %s", i);  }  }  } |

|  |
| --- |
| public class forfor {  public static void main(String args[]){  for(int i = 1, j = 1; i < 100 && j < 100; i++, j = j + 20 ) {  // Have used format instead of println  System.out.format ("\nvalue of i and j are %s and %s ", i, j);  }  }  } |

>> Arrays – Different ways in which an array can be defined

// **int** arr[]={12,23,44,56,78};

int month\_days[];

month\_days = new int[12];

// int month\_days[] = new int[12];

>> Single , Multi dimensional arrays (2D, 3D) – We will cover 2D arrays in some time

>> Length of an array – 1D, 2D

<https://docs.oracle.com/javase/8/docs/api/java/util/Arrays.html>

**Q.? After creating an array, can the size be increased ? NO. Later on, we shall explore ArrayList which allows you to have arrays that can grow.**

First row – index, 2nd row – value for that index

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| A | B | C | D | E | F | G |

Define this array

How would you define this single dimensional array ?

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| “test” | “ab” | “cd” | “beta” | “java” | “x” | “12” | “23” | “os” | “unix” |
|  |  |  |  |  |  |  |  |  |  |

Do this yourself

What about multi dimensional arrays ? Lets visualize 2D .

In Java, multidimensional arrays are actually arrays of arrays

Looping through 2D. Concept of length in 2D arrays.

For loop for Arrays – With and Without counters [covered earlier]

|  |
| --- |
| for(int i = 0; i< arrData.length; i++){  System.out.println(arrData[i]);  } |
| for (String strTemp : arrData){  System.out.println(strTemp);  } |

|  |
| --- |
| class TwoDAgain  { public static void main(String args[])  { int twoD[][] = new int[4][];  twoD[0] = new int[1];  twoD[1] = new int[2];  twoD[2] = new int[3];  twoD[3] = new int[4];  int i, j, k = 0;  for(i=0; i |

**2D Arrays**

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

|  |
| --- |
|  |

|  |
| --- |
| int twoD[][] = new int[4][];  twoD[0] = new int[5];  twoD[1] = new int[5];  twoD[2] = new int[5];  twoD[3] = new int[5]; |

|  |
| --- |
| // Manually allocate differing size second dimensions.  int twoD[][] = new int[4][];  twoD[0] = new int[2];  twoD[1] = new int[3];  twoD[2] = new int[4];  twoD[3] = new int[5]; |

**Alternative Array Declaration Syntax**

There is a second form that may be used to declare an array: type[ ] var-name;

Here, the square brackets follow the type specifier, and not the name of the array variable.

For example, the following two declarations are equivalent:

int al[] = new int[3];

int[] a2 = new int[3];

The following declarations are also equivalent:

char twod1[][] = new char[3][4];

char[][] twod2 = new char[3][4];

**ASSIGNMENTS**

1. For a student, store the marks for say 5/6/… subjects. Use Looping to print the subject name and the marks
2. **conditional Operator**: We normally use the if-then-else statements fr evaluating conditions. Java include a special operator ‘?’ which can replace if-then-else statements for evaluating conditions. The general form of ‘?’ operator is:

*expression1 ? expression2 : expression3*

*Write a simple Java program that uses this operator*

1. 1 = Monday, 2 = Tuesday, ….

Write a program which accepts the day of the week and prints the day as a string

24th onwards

Introduction to OOPS

Q? Have we created classes so far ?

Q. Why doesn’t String require new ?

// Blocks of Code { }