Code Challenge:

WAP to input a number and check if it is prime or not. Write the program using both for loop and while loop.

For Each loop is the advance form of FOR loop. It is very useful while performing iterations through any Array. So let's practice the simple program based on the For Each Loop.

Click on "try yourself" button and get started.

package com.internshala.javaapp;  
  
public class Main {  
  
public static void main(String[] args) {  
  
/\* FOR EACH Loop Practice \*/  
  
// Define a double array and use For Each loop to find the sum of all the numbers stored in the array.  
  
  
}  
  
}

Modify the code that we wrote in the previous video and write a program to find out the sum of all elements present in the 2D Array.

Click on "try yourself" button and try the above mentioned code yourself.

package com.internshala.javaapp;  
public class Main {  
public static void main(String[] args) {  
/\* Define a 2D array with 2 rows and 3 columns and insert int data into it.  
Hint: Check helper text for more details \*/ }  
}

Q. Why are strings immutable?

Strings in Java are immutable most commonly due to these two reasons:

1. Security: Security parameters are typically represented as String in network connection URL's usernames/ passwords, etc. If it were mutable these parameters could be easily changed thus compromising security.

2. Class loading: Strings are used as arguments for class loading. If it were mutable, it could result in the wrong class being loaded as mutable objects tend to change their state.

Let us assume a String  
  
      String myString = "Welcome";

1. Strings are case sensitive i.e. "HELLO" is not equal to "hello". So if you need to compare Strings irresepective of their cases then use  
  
       myString.equalsIgnoreCase("WeLCOme")); // returns true

2. An empty String is represented by  
  
       String str = "" . To check if a String is empty or not use str.isEmpty(); // returns true

3. To get the index of a particular character present in String use  
  
       myString.indexOf('c'); // returns 3

4. To Remove blank space if present at the beginning or at the end of string use  
  
       myString.trim();

String is one of the primitive data types such as int, float, char etc.

* True
* False

Find the output of the following code snippet  
  
String str = "You are geek!";  
System.out.print(str.charAt(8));

* r
* !
* e
* g
* .

Q. What is return type of method?

Return type in a method is the type of result it returns.

e.g. int add(int a, int b){

return a + b;

}

The return statement looks like this: return &lt;A variable or expression yielding a result&gt;;

The result should match with what we write before the function name. That is, a + b should be int as

we have specified.

#### Code Challenge:

WAP to find the sum of two numbers and print the sum by returning the value from the method

public class Main {

public static void main(String[] args) {

/\* WAP for Simple Interest. SI = P \* R \* T / 100 \*/

/\* User 1: principal = 10000, rateOfInterest = 8.7 % , timePeriod = 2 years \*/

double si1 = calculateSimpleInterest(10000, 8.7f, 2);

System.out.println(si1);

/\* User 2: principal = 130000, rateOfInterest = 10.5 % , timePeriod = 4 years \*/

double si2 = calculateSimpleInterest(130000, 10.5f, 4);

System.out.println(si2);

// User 3:

double si3 = calculateSimpleInterest(4000, 12.7f, 8);

System.out.println(si3);

}

public static double calculateSimpleInterest(int principal, float rateOfInterest, int timePeriod) {

double si = principal \* rateOfInterest \* timePeriod / 100;

return si;

}

}

Q. What is the difference between public and private modifiers?

A. When we declare a private method, it is only accessible within a Class. But when we declare a public method, it is not only aceesible within the Class but also thoughout the projects.

Currently, we cannot discuss the differences between public and private modifiers in deep detail. For more information about these two modifiers let's checkout the next module of Object Oriented Programming.

Q. How will we know that our program can have ArrayIndexOutOfBoundsException?

Whenever you are using an array in your program and if you are using the array is of large data storage.

Example: Here the array size is 10 and you are entering at 15th position then ArrayIndexOutOfBoundsException is thrown.

Few commonly encountered Exceptions in Java:  
  
1. Casting: ClassCastException  
2. Arrays: ArrayIndexOutOfBoundsException, NullPointerException  
3. IO: IOException, FileNotFoundException, EOFException  
4. Threads: InterruptedException, SecurityException, IllegalThreadStateException  
  
Potentially common to all situations: NullPointerException, IllegalArgumentException

If you are curious to know more about any of the above mentioned Exceptions then please Google it for your extra knowledge.

Leveraging Basic Concepts

  Module Test Summary

Leveraging Basic Concepts Test Summary

Your Marks 53%

1 Which of these is necessary to specify at time of array initialization?

Attempted answer: None of the mentioned

Correct answer: Row

2 Which loop one should use if the number of iterations are fixed?

Attempted answer: for

3 While loop is generally used when the number of iteration is not fixed.

Attempted answer: True

4 Find the output:  
  
int[][] myArray = {  
    {3, 5, 1, 9 },  
    {10, 15, 3, 0 },   
    {1, 11, 31, 90 },   
    {2, 51, 1, 9 }  
};  
System.out.print(myArray[1][2])

Attempted answer: 3

5 The arrays cannot contain dissimilar data types.

Attempted answer: true

6 Find the output:   
  
int[] arr;  
arr = { 21, 50, 10 };  
System.out.print(arr[1]);

Attempted answer: 21

Correct answer: Error

7 The collection of statements that are grouped together to perform a specific operation is \_\_\_\_\_\_\_\_\_

Attempted answer: Method

8 The concept of method overloading does not depend on \_\_\_\_\_

Attempted answer: method name

Correct answer: the return type

9 Exception handling is used to handle exception during \_\_\_\_\_\_

Attempted answer: compile time

Correct answer: runtime

10 Find the output:  
  
int x = 5;  
  
if (x != 0 )  
{  
int y = x + 1;  
}  
  
System.out.println(y);

Attempted answer: Compilation Error