**LAB 11**

**Implementation of break and continue keyword in java**

**OBJECTIVE**

**To gain familiarity with the Implementation of break and continue keyword in java**

**Hanzala 2023F-BCET-011**

**Exercise**

**TASK 01**

**Objective:** Create a Java program that uses nested loops, the break and continue keywords, and a given integer n to identify the first instance of a prime number bigger than that number

**CODE:**

import java.util.Scanner;

public class PrimeNumber {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.println("Enter an integer n:");

int n = input.nextInt();

int nextPrime = n + 1;

while (true) {

boolean isPrime = true;

for (int i = 2; i <= Math.sqrt(nextPrime); i++) {

if (nextPrime % i == 0) {

isPrime = false;

break;

}

}

if (isPrime) {

System.out.println("The first prime number greater than " + n + " is " + nextPrime);

break;

}

else {

nextPrime++;

continue;

}

}

}

}

**OUTPUT:**

run:

Enter an integer n:

37

The first prime number greater than 37 is 41

BUILD SUCCESSFUL (total time: 13 seconds)

**Hanzala 2023F-BCET-011**

**TASK 02**

**Objective:** Write a Java program to search for a specific character in a given string. If found, print "Character found at index" followed by its index. If not found, print "Character not found”.

**CODE:**

import java.util.Scanner;

public class CharacterSearch {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.println("Enter a string:");

String inputString = input.nextLine();

System.out.println("Enter a character to search for:");

char searchChar = input.next().charAt(0);

boolean found = false;

for (int i = 0; i < inputString.length(); i++) {

if (inputString.charAt(i) == searchChar) {

System.out.println("Character found at index " + i);

found = true;

break;

}

}

if (!found) {

System.out.println("Character not found");

}

}

}

**OUTPUT:**

run:

Enter a string:

Hanzala

Enter a character to search for:

z

Character found at index 3

BUILD SUCCESSFUL (total time: 16 seconds)

**Hanzala 2023F-BCET-011**

**TASK 03**

**Objective:** Write a Java program to print the first 10 numbers of the Fibonacci sequence, skipping numbers that are divisible by 3.

**CODE:**

public class Fibonacci {

public static void main(String[] args) {

int count = 0;

int num1 = 0, num2 = 1;

System.out.println("First 10 Fibonacci numbers:");

while (count < 10) {

int fib = num1 + num2;

num1 = num2;

num2 = fib;

if (fib % 3 == 0) {

continue;

}

System.out.println(fib);

count++;

}

}

}

**OUTPUT:**

run:

First 10 Fibonacci numbers:

1

2

5

8

13

34

55

89

233

377

BUILD SUCCESSFUL (total time: 0 seconds)

**Hanzala 2023F-BCET-011**

**TASK 04**

**Objective:** Write a Java program to find the first non-repeated character in a given string. If all characters are repeated, print "All characters are repeated"

**CODE:**

import java.util.HashMap;

import java.util.Scanner;

public class FirstNonRepeatedCharacter {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.println("Enter a string:");

String inputString = input.nextLine();

HashMap<Character, Integer> charCountMap = new HashMap<>();

for (char c : inputString.toCharArray()) {

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

boolean found = false;

for (char c : inputString.toCharArray()) {

if (charCountMap.get(c) == 1) {

System.out.println("First non-repeated character is: " + c);

found = true;

break;

}

}

if (!found) {

System.out.println("All characters are repeated");

}

}

}  
  
**OUTPUT**:

run:

Enter a string:

Hanzala

First non-repeated character is: H

BUILD SUCCESSFUL (total time: 4 seconds)