CSA09 – Programming in Java

# Day 5 Assignment Questions

SUBMITTED BY:K.HARSHA SAI TEJA REGISTER NO:192111481

1. Develop an Event Handling Applet Program in Java to print a message When the

button is clicked.

# PROGRAM:

import java.applet.\*;

import java.awt.\*;

import java.awt.event.\*;

public class eventhandlingapplet {

Button myButton;

public void init() {

myButton = new Button("Click me!");

myButton.addActionListener(this);

add(myButton);

}

public void actionPerformed(ActionEvent e) {

System.out.println("Button clicked!");

}

}

2. Generate a Java Code to Write and Read the String “WELCOME TO SSE” using

FileOutputStream and FileInputStream class.

Debugging

# PROGRAM:

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class filereadwrite {

public static void main(String[] args) {

String filename = "file.txt";

String welcomeString = "WELCOME TO SSE";

try (FileOutputStream fos = new FileOutputStream(filename)) {

fos.write(welcomeString.getBytes());

System.out.println("Successfully wrote the string to the file.");

} catch (IOException e) {

System.err.println("Failed to write the string to the file.");

e.printStackTrace();

}

try (FileInputStream fis = new FileInputStream(filename)) {

byte[] buffer = new byte[fis.available()];

fis.read(buffer);

String readString = new String(buffer);

System.out.println("Successfully read the string from the file: " + readString);

} catch (IOException e) {

System.err.println("Failed to read the string from the file.");

e.printStackTrace();

}

}

}

3. We define the usage of capitals in a word to be right when one of the following cases holds:

All letters in this word are capitals, like &quot;USA&quot;.

All letters in this word are not capitals, like &quot;leetcode&quot;.

Only the first letter in this word is capital, like &quot;Google&quot;.

Given a string word, return true if the usage of capitals in it is right.

Example 1:

Input: word = &quot;USA&quot;

Output: true

# PROGRAM:

import java.util.\*;

public class capital {

public boolean detectCapitalUse(String word) {

int n = word.length();

int count = 0;

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

if (Character.isUpperCase(c)) {

count++;

}

}

if (count == 0 || count == n || (count == 1 && Character.isUpperCase(word.charAt(0)))) {

return true;

}

return false;

}

}

Example 2:

Input: word = &quot;FlaG&quot;

Output: false

Constraints:

1 &lt;= word.length &lt;= 100

word consists of lowercase and uppercase English letters.

class Solution {

    bool detectCapitalUse(string word) {

    }

}

# PROGRAM:

import java.util.\*;

class capitalsolution {

public boolean detectCapitalUse(String word) {

int n = word.length();

boolean allCaps = true, allLower = true, onlyFirstCap = true;

for (int i = 0; i < n; i++) {

char c = word.charAt(i);

if (Character.isUpperCase(c)) {

allLower = false;

if (i > 0) {

onlyFirstCap = false;

}

} else {

allCaps = false;

if (i == 0) {

onlyFirstCap = false;

}

}

}

return allCaps || allLower || onlyFirstCap;

}

}

4. You are given an array of characters letters that is sorted in non-decreasing order, and a

character target. There are at least two different characters in letters.

Return the smallest character in letters that is lexicographically greater than target. If such a

character does not exist, return the first character in letters.

Example 1:

Input: letters = [&quot;c&quot;,&quot;f&quot;,&quot;j&quot;], target = &quot;a&quot;

Output: &quot;c&quot;

Explanation: The smallest character that is lexicographically greater than &#39;a&#39; in letters is &#39;c&#39;.

# PROGRAM:

import java.util.\*;

class lexicographically{

public char nextGreatestLetter(char[] letters, char target) {

int lo = 0, hi = letters.length - 1;

while (lo <= hi) {

int mid = lo + (hi - lo) / 2;

if (letters[mid] <= target) {

lo = mid + 1;

} else {

hi = mid - 1;

}

}

return lo < letters.length ? letters[lo] : letters[0];

}

Example 2:

Input: letters = [&quot;c&quot;,&quot;f&quot;,&quot;j&quot;], target = &quot;c&quot;

Output: &quot;f&quot;

Explanation: The smallest character that is lexicographically greater than &#39;c&#39; in letters is &#39;f&#39;.

# PROGRAM:

import java.util.\*;

class lexicographically1 {

public static char nextGreatestLetter(char[] letters, char target) {

int n = letters.length;

int left = 0, right = n - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (letters[mid] <= target) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return left < n ? letters[left] : letters[0];

}

Example 3:

Input: letters = [&quot;x&quot;,&quot;x&quot;,&quot;y&quot;,&quot;y&quot;], target = &quot;z&quot;

Output: &quot;x&quot;

Explanation: There are no characters in letters that is lexicographically greater than &#39;z&#39; so we

return letters[0].

Constraints:

2 &lt;= letters.length &lt;= 104

letters[i] is a lowercase English letter.

letters is sorted in non-decreasing order.

letters contains at least two different characters.

target is a lowercase English letter.

class Solution {

    char nextGreatestLetter(vector&lt;char&gt;&amp; letters, char target) {

    }

}

# PROGRAM:

import java.util.\*;

class lexicographicallysolution{

public char nextGreatestLetter(char[] letters, char target) {

int n = letters.length;

int lo = 0, hi = n - 1;

while (lo <= hi) {

int mid = lo + (hi - lo) / 2;

if (letters[mid] <= target) {

lo = mid + 1;

} else {

hi = mid - 1;

}

}

return lo < n ? letters[lo] : letters[0];

}

}

5. Program to show syntax of conditional and looping statement by menu choice : Find/Debug

erron in following code

import java.util.Scanner;

public class Menusel

{

public static void main(String args[])

{

Scanner scan = new Scanner(System.in);

charchoice;

do

{

System.out.println(Help on : &quot;);

System.out.println(&quot;1. if&quot;);

System.out.println(&quot;2. switch&quot;);

System.out.println(&quot;3. while&quot;);

System.out.println(&quot;4. do-while&quot;);

Syste.out.println(&quot;5. for\n&quot;);

System.out.println(&quot;Choose any one : &quot;);

choice = scan.next().charAt(0);

}while(choice &lt; &#39;1&#39; &amp;&amp; choice &gt; &#39;5&#39;);

System.out.println(&quot;\n&quot;);

switch(choice)

{

case 1&#39; : System.out.println(&quot;The if :\n&quot;);

System.out.println(&quot;if(condition)\n{\n\tstatement\n}&quot;);

System.out.println(&quot;else\n{\n\tstatement\n}&quot;);

break;

case &#39;2&#39; : System.out.println(&quot;The switch :\n&quot;);

System.out.println(&quot;switch(expression)\n{&quot;);

System.out.println(&quot;\tcase constant: statement sequence\n\tbreak;&quot;);

System.out.println(&quot;\t//...\n}&quot;);

break;

case &#39;3&#39; : System.out.println(&quot;The while :\n&quot;);

System.out.println(&quot;while(condition)\n{&quot;);

System.out.println(&quot;\t// body of loop\n}&quot;);

break;

case &#39;4&#39; : System.out.println(&quot;The do-while :\n&quot;);

System.out.println(&quot;do\n{&quot;);

System.out.println(&quot;\t// body of loop\n\n}while(condition);&quot;);

case &#39;5 : System.out.println(&quot;The for :\n&quot;);

System.out.println(&quot;for(initialization; condition; iteration)\n{&quot;);

System.out.println(&quot;\t// body of loop\n}&quot;);

break;

}

}

}

# PROGRAM:

import java.util.Scanner;

public class Menusel {

public static void main(String args[]) {

Scanner scan = new Scanner(System.in);

char choice;

do {

Syste

System.out.println("1. if");

System.out.println("2. switch");

System.out.println("3. while");

System.out.println("4. do-while");

System.out.println("5. for\n");

System.out.println("Choose any one: ");

choice = scan.next().charAt(0);

} while(choice < '1' || choice > '5');

System.out.println("\n");

switch(choice) {

case '1' :

System.out.println("The if:\n");

System.out.println("if(condition) {\n\tstatement\n}");

System.out.println("else {\n\tstatement\n}");

break;

case '2' :

System.out.println("The switch:\n");

System.out.println("switch(expression) {\n");

System.out.println("\tcase constant: statement sequence\n\tbreak;");

System.out.println("\t//...\n}");

break;

case '3' :

System.out.println("The while:\n");

System.out.println("while(condition) {\n");

System.out.println("\t// body of loop\n}");

break;

case '4' :

System.out.println("The do-while:\n");

System.out.println("do {\n");

System.out.println("\t// body of loop\n} while(condition);");

break;

case '5' :

System.out.println("The for:\n");

System.out.println("for(initialization; condition; iteration) {\n");

System.out.println("\t// body of loop\n}");

break;

default :

System.out.println("Invalid choice.");

}

}

}

m.out.println("Help on:");

6. Write a program to count all the prime and composite numbers entered by the user.

Sample Input:

Enter the numbers

4

54

29

71

7

59

98

23

Sample Output:

Composite number:3

Prime number:5

Test cases:

1. 33, 41, 52, 61,73,90

2. TEN, FIFTY, SIXTY-ONE, SEVENTY-SEVEN, NINE

3. 45, 87, 09, 5.0 ,2.3, 0.4

4. -54, -76, -97, -23, -33, -98

5. 45, 73, 00, 50, 67, 44

# PROGRAM:

import java.util.Scanner;

public class PrimeCompositeCounter {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int compositeCount = 0;

int primeCount = 0;

System.out.println("Enter the numbers: ");

while (sc.hasNextInt()) {

int n = sc.nextInt();

if (n < 2) { // 0 and 1 are neither prime nor composite

continue;

}

boolean isComposite = false;

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

if (n % i == 0) {

isComposite = true;

break;

}

}

if (isComposite) {

compositeCount++;

} else {

primeCount++;

}

}

System.out.println("Composite number: " + compositeCount);

System.out.println("Prime number: " + primeCount);

sc.close();

}

}

7. Find the M th maximum number and N th minimum number in an array and then find the

sum of it and difference of it.

Sample Input:

Array of elements = {14, 16, 87, 36, 25, 89, 34}

M = 1

N = 3

Sample Output:

1 st Maximum Number = 89

3 rd Minimum Number = 25

Sum = 114

Difference = 64

Test cases:

1. {16, 16, 16 16, 16}, M = 0, N = 1

2. {0, 0, 0, 0}, M = 1, N = 2

3. {-12, -78, -35, -42, -85}, M = 3 , N = 3

4. {15, 19, 34, 56, 12}, M = 6 , N = 3

5. {85, 45, 65, 75, 95}, M = 5 , N = 7

# PROGRAM:

import java.util.Arrays;

public class ArrayMinMaxSumDiff {

public static void main(String[] args) {

int[] arr = {14, 16, 87, 36, 25, 89, 34};

int m = 1;

int n = 3;

Arrays.sort(arr);

int mthMax = arr[arr.length - m];

int nthMin = arr[n - 1];

int sum = mthMax + nthMin;

int diff = mthMax - nthMin;

System.out.println("Mth maximum number = " + mthMax);

System.out.println("Nth minimum number = " + nthMin);

System.out.println("Sum = " + sum);

System.out.println("Difference = " + diff);

}

}

8. In an organization they decide to give bonus to all the employees on New Year. A 5%

bonus on salary is given to the grade A workers and 10% bonus on salary to the grade

B workers. Write a program to enter the salary and grade of the employee. If the

salary of the employee is less than $10,000 then the employee gets an extra 2% bonus

on salary Calculate the bonus that has to be given to the employee and print the salary

that the employee will get.

Sample Input &amp; Output:

Enter the grade of the employee: B

Enter the employee salary: 50000

Salary=50000

Bonus=5000.0

Total to be paid:55000.0

Test cases:

1. Enter the grade of the employee: A

Enter the employee salary: 8000

2. Enter the grade of the employee: C

Enter the employee salary: 60000

3. Enter the grade of the employee: B

Enter the employee salary: 0

4. Enter the grade of the employee: 38000

Enter the employee salary: A

5. Enter the grade of the employee: B

Enter the employee salary: -8000

# PROGRAM:

import java.util.Scanner;

public class BonusCalculator {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter the grade of the employee: ");

String grade = input.next();

System.out.print("Enter the employee salary: ");

double salary = input.nextDouble();

double bonus = 0.0;

if (salary < 10000) {

bonus += salary \* 0.02;

}

if (grade.equals("A")) {

bonus += salary \* 0.05;

} else if (grade.equals("B")) {

bonus += salary \* 0.1;

}

double totalSalary = salary + bonus;

System.out.println("Salary=" + salary);

System.out.println("Bonus=" + bonus);

System.out.println("Total to be paid:" + totalSalary);

}

}

9. Write a program to print the first n perfect numbers. (Hint Perfect number means a

positive integer that is equal to the sum of its proper divisors)

Sample Input:

N = 3

Sample Output:

First 3 perfect numbers are: 6 , 28 , 496

Test Cases:

1. N = 0

2. N = 5

3. N = -2

4. N = -5

5. N = 0.2

# PROGRAM:

import java.util.ArrayList;

public class PerfectNumbers {

public static void main(String[] args) {

int n = 3; // replace with desired value of n

ArrayList<Integer> perfectNumbers = new ArrayList<>();

int i = 1;

while (perfectNumbers.size() < n) {

int sumOfDivisors = 0;

for (int j = 1; j < i; j++) {

if (i % j == 0) {

sumOfDivisors += j;

}

}

if (sumOfDivisors == i) {

perfectNumbers.add(i);

}

i++;

}

System.out.print("First " + n + " perfect numbers are: ");

for (int perfectNumber : perfectNumbers) {

System.out.print(perfectNumber + " ");

}

}

}

10. Write a program to enter the marks of a student in four subjects. Then calculate the total

and aggregate, display the grade obtained by the student. If the student scores an

aggregate greater than 75%, then the grade is Distinction. If aggregate is 60&gt;= and &lt;75,

then the grade is First Division. If aggregate is 50 &gt;= and &lt;60, then the grade is Second

Division. If aggregate is 40&gt;= and &lt;50, then the grade is Third Division. Else the grade is

Fail.

Sample Input &amp; Output:

Enter the marks in python: 90

Enter the marks in c programming: 91

Enter the marks in Mathematics: 92

Enter the marks in Physics: 93

Total= 366

Aggregate = 91.5

DISTINCTION

Test cases:

a) 18, 76,93,65

b) 73,78,79,75

c) 98,106,120,95

d) 96,73, -85,95

e) 78,59.8,76,79

# PROGRAM:

import java.util.Scanner;

public class StudentMarks {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter the marks in python: ");

int python = input.nextInt();

System.out.print("Enter the marks in c programming: ");

int cProgramming = input.nextInt();

System.out.print("Enter the marks in Mathematics: ");

int mathematics = input.nextInt();

System.out.print("Enter the marks in Physics: ");

int physics = input.nextInt();

int total = python + cProgramming + mathematics + physics;

double aggregate = total / 4.0;

System.out.println("Total= " + total);

System.out.println("Aggregate = " + aggregate);

if (aggregate >= 75) {

System.out.println("DISTINCTION");

} else if (aggregate >= 60) {

System.out.println("FIRST DIVISION");

} else if (aggregate >= 50) {

System.out.println("SECOND DIVISION");

} else if (aggregate >= 40) {

System.out.println("THIRD DIVISION");

} else {

System.out.println("FAIL");

}

}

}

11. Write a program to read the numbers until -1 is encountered. Find the average of positive

numbers and negative numbers entered by user.

Sample Input:

Enter -1 to exit…

Enter the number: 7

Enter the number: -2

Enter the number: 9

Enter the number: -8

Enter the number: -6

Enter the number: -4

Enter the number: 10

Enter the number: -1

Sample Output:

The average of negative numbers is: -5.0

The average of positive numbers is : 8.66666667

Test cases:

1. -1,43, -87, -29, 1, -9

2. 73, 7-6,2,10,28,-1

3. -5, -9, -46,2,5,0

4. 9, 11, -5, 6, 0,-1

5. -1,-1,-1,-1,-1

# PROGRAM:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int num, sumPositive = 0, sumNegative = 0, countPositive = 0, countNegative = 0;

System.out.println("Enter -1 to exit...");

while (true) {

System.out.print("Enter the number: ");

num = sc.nextInt();

if (num == -1) {

break;

} else if (num > 0) {

sumPositive += num;

countPositive++;

} else {

sumNegative += num;

countNegative++;

}

}

if (countPositive > 0) {

double avgPositive = (double) sumPositive / countPositive;

System.out.printf("The average of positive numbers is: %.2f\n", avgPositive);

} else {

System.out.println("No positive numbers entered.");

}

if (countNegative > 0) {

double avgNegative = (double) sumNegative / countNegative;

System.out.printf("The average of negative numbers is: %.2f\n", avgNegative);

} else {

System.out.println("No negative numbers entered.");

}

}

}

12. Write a program to read a character until a \* is encountered. Also count the number of

uppercase, lowercase, and numbers entered by the users.

Sample Input:

Enter \* to exit…

Enter any character: W

Enter any character: d

Enter any character: A

Enter any character: G

Enter any character: g

Enter any character: H

Enter any character: \*

Sample Output:

Total count of lower case:2

Total count of upper case:4

Total count of numbers =0

Test cases:

1. 1,7,6,9,5

2. S, Q, l, K,7, j, M

3. M, j, L, &amp;, @, G

4. D, K, I, 6, L, \*

5. \*, K, A, e, 1, 8, %, \*

# PROGRAM:

import java.util.Scanner;

public class CharacterCount {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int uppercaseCount = 0;

int lowercaseCount = 0;

int numberCount = 0;

System.out.println("Enter \* to exit...");

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

while (ch != '\*') {

if (Character.isUpperCase(ch)) {

uppercaseCount++;

} else if (Character.isLowerCase(ch)) {

lowercaseCount++;

} else if (Character.isDigit(ch)) {

numberCount++;

}

ch = input.next().charAt(0);

}

System.out.println("Total count of lower case:" + lowercaseCount);

System.out.println("Total count of upper case:" + uppercaseCount);

System.out.println("Total count of numbers =" + numberCount);

}

}

13. Using the concepts of thread with implementing Runnable interface in Java to find

whether a given number is prime or not.

Sample Input : 5

Sample Output : 5 is Prime

Sample Output : 15

Test Cases

1. 4

2. -10

3. 0

4. EIGHT SEVEN

5. 11.48

# PROGRAM:

public class PrimeChecker implements Runnable {

private int number;

public PrimeChecker(int number) {

this.number = number;

}

public void run() {

boolean isPrime = true;

if (number <= 1) {

isPrime = false;

} else {

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

if (number % i == 0) {

isPrime = false;

break;

}

}

}

if (isPrime) {

System.out.println(number + " is prime.");

} else {

System.out.println(number + " is not prime.");

}

}

public static void main(String[] args) {

PrimeChecker pc1 = new PrimeChecker(4);

PrimeChecker pc2 = new PrimeChecker(-10);

PrimeChecker pc3 = new PrimeChecker(0);

PrimeChecker pc4 = new PrimeChecker(87);

PrimeChecker pc5 = new PrimeChecker(11);

Thread t1 = new Thread(pc1);

Thread t2 = new Thread(pc2);

Thread t3 = new Thread(pc3);

Thread t4 = new Thread(pc4);

Thread t5 = new Thread(pc5);

t1.start();

t2.start();

t3.start();

t4.start();

t5.start();

}

}

14. Bring out the situation in which member names of a subclass hide members by the

same name in the super class. How it can be resolved? Write Suitable code in Java

and

Implement above scenario with the Parametrized Constructor (accept int type

parameter) of the Super Class can be called from Sub Class Using super () and display

the input values provided.

Sample Input : 100, 200

Sample Output : 100, 200

Test Cases

1. 10, 20

2. -20, -30

3. 0, 0

4. EIGHT FIVE

5. 10.57, 12.58

# PROGRAM:

class SuperClass {

int num;

public SuperClass(int num) {

this.num = num;

}

public void display() {

System.out.println("Number in SuperClass: " + num);

}

}

class SubClass extends SuperClass {

int num;

public SubClass(int num1, int num2) {

super(num1);

this.num = num2;

}

public void display() {

super.display();

System.out.println("Number in SubClass: " + num);

}

}

public class Main {

public static void main(String[] args) {

SubClass obj = new SubClass(100, 200);

obj.display();

}

}

15. Display Multiplication table for 5 and 10 using various stages of life cycle of the

thread by generating a suitable code in Java.

Sample Input 5, 10

5 X 1 = 5

5 X 2 =10

….

10 X 1 =10

10 X 2 = 20

….

Test Cases:

1. 10, 20

2. -10, -30

3. 0, 0

4. SIX, SIX

9.8, 9.6

# PROGRAM:

public class MultiplicationTable implements Runnable {

private int number;

public MultiplicationTable(int number) {

this.number = number;

}

public void run() {

for (int i = 1; i <= 10; i++) {

System.out.println(number + " X " + i + " = " + (number \* i));

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

public static void main(String[] args) {

MultiplicationTable table1 = new MultiplicationTable(5);

MultiplicationTable table2 = new MultiplicationTable(10);

Thread t1 = new Thread(table1);

Thread t2 = new Thread(table2);

t1.start();

t2.start();

try {

t1.join();

t2.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("Multiplication tables displayed successfully!");

}

}