Lab Assignment 04

User Input and Advanced Loops



CSE110: Programming Language I

| **No of Tasks** | | | **Points to Score** |
| --- | --- | --- | --- |
| **15** | | | **150** |

Submit the coding tasks (Task 1 - 11) on buX and the handwritten tasks (Task 12 - 15) to your Lab Instructors in the beginning of the next lab class.

1. Write a Java program that asks the user how many inputs they want to provide and then takes that many inputs and prints the maximum, minimum, and average of all the even positive numbers given by the user. If no even positive number is given, the average should be zero.

| **Sample Input** | **Sample Output** |
| --- | --- |
| 5  12  -8  19  8  -1 | Max: 12  Min: 8  Average: 10 |
| **Explanation:**  At first the user gave 5 as the input which indicates that the user will provide 5 numbers. Then 5 numbers were taken as inputs. Among these, only 12 and 8 are even positive numbers. | |

import java.util.Scanner;

public class Task01{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Ask the user how many inputs they want to provide

System.out.print("How many numbers do you want to input? ");

int count = sc.nextInt();

// Variables to store max, min, and sum of even positive numbers

int max = -1;

int min = 1;

int sum = 0;

int evenCount = 0;

// Loop to collect the inputs

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

System.out.print("Enter number " + (i + 1) + ": ");

int number = sc.nextInt();

// Check if the number is even and positive

if (number > 0 && number % 2 == 0) {

// Update max and min

if (number > max) {

max = number;

}

if (number < min) {

min = number;

}

// Update sum and count of even positive numbers

sum += number;

evenCount++;

}

}

// Calculate average

double average = (evenCount > 0) ? (double) sum / evenCount : 0;

// Print results

if (evenCount > 0) {

System.out.println("Max: " + max);

System.out.println("Min: " + min);

System.out.printf("Average: "+ average);

} else {

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

System.out.println("Max: N/A");

System.out.println("Min: N/A");

System.out.println("Average: 0.00");

}

// Close the scanner

sc.close();

}

}

1. Write a Java program that will keep taking integer numbers as inputs from the user and print the square of those numbers until it gets a negative number and then stop.

**Sample Input/Output:** (The purple numbers are input.)

Enter Number: 2

2 ^ 2 = 4

Enter Number: 6

6 ^ 2 = 36

Enter Number: 1

1 ^ 2 = 1

Enter Number: 4

4 ^ 2 = 16

Enter Number: -5

import java.util.Scanner;

public class Task02 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int number;

while (true) {

System.out.print("Enter Number: ");

number = scanner.nextInt();

// Check if the number is negative

if (number < 0) {

break; // Exit the loop if the number is negative

}

// Calculate the square and print the result

int square = number \* number;

System.out.println(number + " ^ 2 = " + square);

}

System.out.println("Program terminated. You entered a negative number.");

scanner.close();

}

}

1. Write a Java code that asks an integer as input from the user and takes that many integer inputs. Your task is to count how many numbers are non-negative and negative.

**Sample Input:** (The purple numbers are input.)

Enter an integer: 9

Enter number 1: -8

Enter number 2: 33

Enter number 3: -100

Enter number 4: 10

Enter number 5: 0

Enter number 6: 5

Enter number 7: 10

Enter number 8: -4

Enter number 9: 4

**Sample Output:**

6 Non-negative Numbers

3 Negative Numbers

import java.util.Scanner;

public class Task03 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ask for the number of inputs

System.out.print("Enter an integer: ");

int count = scanner.nextInt();

int nonNegativeCount = 0;

int negativeCount = 0;

// Loop to take the specified number of inputs

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

System.out.print("Enter number " + i + ": ");

int number = scanner.nextInt();

// Count non-negative and negative numbers

if (number < 0) {

negativeCount++;

} else {

nonNegativeCount++;

}

}

// Display the results

System.out.println(nonNegativeCount + " Non-negative Numbers");

System.out.println(negativeCount + " Negative Numbers");

// Close the scanner

scanner.close();

}

}

1. Write a Java program to take a positive integer ***N*** (where N > 0) as user input and print the **first *N* prime numbers starting from 2**. Your code should check all the positive integers starting from 2 and determine whether they are prime or not until ***N*** prime numbers are found.

**Sample Input 1:**

5

**Sample Output 1:**

2

3

5

7

11

**Sample Input 2:**

7

**Sample Output 2:**

2

3

5

7

11

13

17

import java.util.Scanner;

public class Task04v2{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Prompt user for a positive integer N

System.out.print("Enter a positive integer N (N > 0): ");

int N = scanner.nextInt();

int count = 0; // Count of prime numbers found

int number = 2; // Starting number to check for primality

// Loop until we find N prime numbers

while (count < N) {

int c=0; // Assume the number is prime

// Check if 'number' is prime

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

if (number % i == 0) {

c++; // No need to check further

}

}

// If the number is prime, print it

if (c==1) {

System.out.println(number);

count++; // Increment the count of found primes

}

number++; // Move to the next number

}

}

}

1. Write a Java code of a program that reads the value of N (where N > 0) from the user and calculates the value of y if the expression of y is as follows:

**Sample Input:**

The value of N: 2

**Sample Output:**

The value of y: -4

**Sample Input:**

The value of N: 4

**Sample Output:**

The value of y: -20

import java.util.Scanner;

public class Task05 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Prompt user for a positive integer N

System.out.print("The value of N: ");

int N = scanner.nextInt();

int y = 0; // Initialize y

// Calculate the value of y based on the given expression

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

// Calculate the sum of the first i integers

int sum = (i \* (i + 1)) / 2; // Using the formula for the sum of the first i natural numbers

y -= sum; // Subtract the sum from y

}

System.out.println("The value of y: " + y);

}

}

1. Write a Java program that will keep taking even positive integer numbers as inputs from the user and print the number of divisors of those numbers until it gets an odd number and then stops.

**Sample Input & Output:** (The purple numbers are input)

Enter Number: 44

44 has 6 divisors

Enter Number: 30

30 has 8 divisors

Enter Number: 8

8 has 4 divisors

Enter Number: 4

4 has 3 divisors

Enter Number: 6

6 has 4 divisors

Enter Number: 20

20 has 6 divisors

Enter Number: 5

import java.util.Scanner;

public class Task06 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.print("Enter Number: ");

int number = scanner.nextInt();

// Check if the number is odd

if (number % 2 != 0) {

break; // Exit the loop if the number is odd

}

// Calculate the number of divisors

int divisorCount = 0;

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

if (number % i == 0) {

divisorCount++; // Increment the count of divisors

}

}

System.out.println(number + " has " + divisorCount + " divisors");

}

}

}

1. Read an integer N that is the number of test cases that follow. Each test case contains two integers X and Y. Print one output line for each test case that the sum of Y odd numbers from X including it if is the case. For example:

For the input 4 5, the output must be 45, that is: 5 + 7 + 9 + 11 + 13

For the input 7 4, the output must be 40, that is: 7 + 9 + 11 + 13

| **Sample Input** | **Sample Output** |
| --- | --- |
| 2  4  3  11  2 | 21  24 |
| Explanation: Here, the 2 means there are two test cases. For each test case you have to take two inputs (X, Y) and print the sum of Y odd numbers starting from X. | |

import java.util.Scanner;

public class Task07{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int N = scanner.nextInt();

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

// Read X and Y for each test case

System.out.print("Enter X and Y: ");

int X = scanner.nextInt();

int Y = scanner.nextInt();

int sum = 0; // Variable to store the sum of odd numbers

int count = 0; // Counter for how many odd numbers have been added

// Start from X and find Y odd numbers

while (count < Y) {

if (X % 2 != 0) { // Check if X is odd

sum += X; // Add to sum if odd

count++; // Increment the count of odd numbers found

}

X++; // Move to the next number

}

// Print the result for the current test case

System.out.println("Sum of " + Y + " odd numbers starting from " + X + ": " + sum);

}

}

}

1. Take the length and width of a **rectangle** from the user and create the rectangle according to the output below. Your output should match the specified output.

| **Sample Input #1**  4  6 | **Sample Input #2**  3  5 |
| --- | --- |
| **Output**  1 2 3 4  1 2 3 4  1 2 3 4  1 2 3 4  1 2 3 4  1 2 3 4 | **Output**  1 2 3  1 2 3  1 2 3  1 2 3  1 2 3 |

import java.util.Scanner;

public class Task08{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the width of the rectangle: ");

int col = scanner.nextInt();

System.out.print("Enter the length of the rectangle: ");

int row = scanner.nextInt();

// Print the rectangle

for (int i = 0; i < row; i++) { // Loop for each row

for (int j = 1; j <= col; j++) { // Loop for each column

System.out.print(j + " "); // Print numbers from 1 to width

}

System.out.println(); // Move to the next line after each row

}

}

}

1. Take the height of a **right-justified right triangle** from the user and create the triangle according to the output below. Your output should match the specified output.

| **Sample Input #1**  4 | **Sample Input #2**  3 |
| --- | --- |
| **Output**  1  1 2  1 2 3  1 2 3 4 | **Output**  1  1 2  1 2 3 |

import java.util.Scanner;

public class Task09{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Read the height of the triangle from the user

System.out.print("Enter the height of the triangle: ");

int height = scanner.nextInt();

// Print the right-justified triangle

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

// Print leading spaces for right justification

for (int j = 0; j < height - i; j++) {

System.out.print(" "); // Two spaces for each leading space

}

// Print numbers from 1 to i

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

System.out.print(j + " "); // Print the numbers followed by a space

}

// Move to the next line after each row

System.out.println();

}

}

}

1. Take the height of an **isosceles triangle** from the user and create the triangle according to the output below. Your output should match the specified output.

| **Sample Input #1**  4 | **Sample Input #2**  3 |
| --- | --- |
| **Output**  1  1 2 3  1 2 3 4 5  1 2 3 4 5 6 7 | **Output**  1  1 2 3  1 2 3 4 5 |

import java.util.Scanner;

public class Task10 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Read the height of the triangle from the user

System.out.print("Enter the height of the triangle: ");

int height = scanner.nextInt();

// Print the isosceles triangle

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

// Print leading spaces for center alignment

for (int j = 0; j < height - i; j++) {

System.out.print(" "); // Two spaces for each leading space

}

// Print numbers from 1 to (2 \* i - 1)

for (int j = 1; j <= (2 \* i - 1); j++) {

System.out.print(j + " "); // Print the numbers followed by a space

}

// Move to the next line after each row

System.out.println();

}

}

}

1. Write a Java program that will ask for a range (a starting number and an ending number) from the user and print all the Armstrong numbers between that range.

***[Armstrong Number:****An Armstrong number is a number whose sum of digits raised to the power the number of digits equals to that number.*

*For example, 371 is an Armstrong number because 33 + 73 + 13 = 371, here the total number of digits in 371 is 3* ***]***

**Sample Input 1:**

Start: 300

End: 500

**Sample Output 1:**

Armstrong numbers:

370 ->

371

407

**Sample Input 2:**

Start: 100

End: 200

**Sample Output 2:**

Armstrong numbers:

153

# 

import java.util.Scanner;

public class Task11 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Prompt user for the starting and ending numbers

System.out.print("Start: ");

int start = scanner.nextInt();

System.out.print("End: ");

int end = scanner.nextInt();

System.out.println("Armstrong numbers: ");

// Loop through the range and check for Armstrong numbers

for (int number = start; number <= end; number++) {

int originalNumber = number;

int sum = 0;

int numberOfDigits = String.valueOf(number).length();

// Calculate the sum of the digits raised to the power of the number of digits

while (number > 0) {

int digit = number % 10;

sum += Math.pow(digit, numberOfDigits);

number /= 10;

}

// Check if the calculated sum is equal to the original number

if (sum == originalNumber) {

System.out.println(originalNumber);

}

// Reset the number for the next iteration

number = originalNumber; // Restore the original number for the next loop iteration

}

}

}

1. Trace the following code, create a tracing table and write the outputs.

| **1** | **public class T1{** |
| --- | --- |
| **2** | **public static void main(String args[]){** |
| **3** | **int x = 0, y = 0;** |
| **4** | **int sum = 0;** |
| **5** | **while (x < 4){** |
| **6** | **y = x - 3;** |
| **7** | **while (y < 3){** |
| **8** | **sum = (sum % 3) + x - y \* 3 ;** |
| **9** | **System.out.println(sum);** |
| **10** | **y = y + 1;** |
| **11** | **}** |
| **12** | **if (x > 5){** |
| **13** | **x++;** |
| **14** | **}** |
| **15** | **else{** |
| **16** | **x += 2;** |
| **17** | **}** |
| **18** | **}** |
| **19** | **}** |
| **20** | **}** |

1. Trace the following code, create a tracing table and write the outputs.

| **1** | **public class T2 {** |
| --- | --- |
| **2** | **public static void main(String args[]) {** |
| **3** | **int x = 0, i = 0, sum = 0;** |
| **4** | **i = 1;** |
| **5** | **x = 2;** |
| **6** | **sum = 0;** |
| **7** | **while (i < 20){** |
| **8** | **x = x + i;** |
| **9** | **sum = sum + x + 1;** |
| **10** | **System.out.println(sum);** |
| **11** | **if (x > 5){** |
| **12** | **i += 2;** |
| **13** | **}** |
| **14** | **else {** |
| **15** | **i += 3;** |
| **16** | **}** |
| **17** | **}** |
| **18** | **sum = sum + i;** |
| **19** | **System.out.println(sum);** |
| **20** | **}** |
| **21** | **}** |

1. Trace the following code, create a tracing table and write the outputs.

| **1** | **public class T3** |
| --- | --- |
| **2** | **{** |
| **3** | **public static void main(String args[])** |
| **4** | **{** |
| **5** | **int x = 0, y = 0;** |
| **6** | **int sum = 0;** |
| **7** | **while (x < 10){** |
| **8** | **y = x - 3;** |
| **9** | **y = 40;** |
| **10** | **while (y > 22){** |
| **11** | **if ((sum > 30) && (sum < 40)){** |
| **12** | **sum = sum + x \* 2 ;** |
| **13** | **}** |
| **14** | **else if ((sum > 40) && (sum < 50)){** |
| **15** | **sum = sum + x \* 3;** |
| **16** | **}** |
| **17** | **else {** |
| **18** | **sum = sum + 23;** |
| **19** | **}** |
| **20** | **System.out.println(sum);** |
| **21** | **y = y - 10;** |
| **22** | **}** |
| **23** | **x += 2;** |
| **24** | **}** |
| **25** | **}** |
| **26** | **}** |

1. Trace the following code, create a tracing table and write the outputs.

| **1** | **public class T4{** |
| --- | --- |
| **2** | **public static void main(String args[]){** |
| **3** | **boolean check = true;** |
| **4** | **int x = 2, y = 2, z = 3;** |
| **5** | **while(check){** |
| **6** | **y = 4 / x % 3 + z \* y - 5;** |
| **7** | **if(y > 10 || x==7){** |
| **8** | **z += 3;** |
| **9** | **break;** |
| **10** | **}** |
| **11** | **if(4+x%3 > 5){** |
| **12** | **x %= y + (z--) + z;** |
| **13** | **System.out.println(x);** |
| **14** | **}** |
| **15** | **else{** |
| **16** | **y += x + (--z) + y;** |
| **17** | **System.out.println(y);** |
| **18** | **}** |
| **19** | **x++;** |
| **20** | **System.out.println(x + y);** |
| **21** | **}** |
| **22** | **}** |
| **23** | **}** |