Lab Assignment 01

|  |  |
| --- | --- |
| **Course Code:** | **CSE111** |
| **Course Title:** | **Programming Language II** |
| **Topic:** | **Loops, String, Arrays** |
| **Number of Tasks:** | **10** |

**Task 1**

Write a java program that takes 10 inputs from the user in a loop, and displays the sum, average, minimum and maximum of **Only the positive odd numbers** from those numbers. If no such numbers are found, then display the message “No odd positive numbers found”.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 1  4  2  9  2  -4  3  -1  0  1 | Sum = 14  Minimum = 1  Maximum = 9  Average = 3.5 |
| 34  -11  50  24  -24  2  -4  0  8  12 | No odd positive numbers found |
| 23  2  -4  0  8  12  34  -11  53  21 | Sum = 97  Minimum = 21  Maximum = 53  Average = 32.333333333333336 |

**Task 2**

Write a java program that takes 2 integer numbers as input and calculates how many prime numbers exist between them.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 10  15 | There are 2 prime numbers between 10 and 15. |
| 150  100 | There are 10 prime numbers between 100 and 150. |

**Task 3**

Write a Java program that takes TWO string inputs (containing exactly one word in each string) from the user. Concatenate those two strings with a single space in between them. Generate a number **which is the sum of all the letters in that concatenated string** where A = 65, Z = 90, a = 97, and z = 122. Your task is to print that concatenated string and the number generated from that string.

|  |  |
| --- | --- |
| **Sample Input** | **Output** |
| Hello123  Wo%%rld | Hello123 Wo%%rld  1020 |
| Ja12-va  CHOWD+ HURY | Ja12-va CHOWD+ HURY  1087 |

**Task 4**

Write a Java program that takes a string input in small letters from the user and prints the previous alphabet in sequence for each alphabet found in the input.

|  |  |
| --- | --- |
| **Sample Input** | **Output** |
| wxyz | vwxy |
| thecow | sgdbnv |
| abcd | zabc |

**Task 5**

Write a Java program that asks the user for the length of an array and then creates an integer array of that length by taking inputs from the user. Then, reverse the **original array** **without** creating any new array and print it. **[In-place reverse]**

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| Enter the length of the array: 5  7  -31  344  97  100 | 100 97 344 -31 7 |

**Task 6**

Write a Java program that will take an integer number N from the user and create an integer array by taking N numbers from the user. Print how many times each number appears in the array.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| N = 5  6  15  14  15  6 | 6 - 2 times  15 - 2 times  14 - 1 times |
| N = 6  -5  10  14  10  -7  10 | -5 - 1 times  10 - 3 times  14 - 1 times  -7 - 1 times |

**Task 7**

Write a Java program that asks the user the length of an array (N) then takes N number of doubles as elements for the array as input. First, remove the consecutive duplicate elements from the original array **to form a new array**. Then print the number of elements removed from the original array.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| N = 8  Please enter the elements of the array:  5.2  2.7  1.0  1.0  2.7  3.5  3.5  3.5 | New Array: 5.2 2.7 1.0 2.7 3.5  Removed elements : 3 |

**Task 8**

Write a Java program that will take the number of rows and columns from the user and create a 2D array by taking integer numbers from the user. Print the 2D array. Finally, create a 1D array by flattening the 2D array.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| row = 2  column = 3  1  2  3  4  5  6 | 2D Array:  1 2 3  4 5 6  1D Array:  1 2 3 4 5 6 |
| row = 3  column = 2  1  4  5  6  8  9 | 2D Array:  1 4  5 6  8 9  1D Array:  1 4 5 6 8 9 |

**Task 9**

You are given a square matrix **A** of size N×N. Check whether the given matrix is an Identity matrix or not. If it is, then print "Identity matrix" or otherwise print "Not an Identity matrix". **Your program should work for any given 2D Array of size N×N.**

[You may need to use the concept of flag and break to solve this problem.]

*Identity Matrix is a square matrix with 1’s along the diagonal from upper left to lower right and 0’s in all other positions.*

|  |  |
| --- | --- |
| **Given Array** | **Output** |
| int [ ] [ ] A = {{1, 0, 0, 1},  {0, 1, 0, 0},  {1, 0, 1, 0},  {0, 1, 0, 1}}; | Not an Identity Matrix |
| int [ ] [ ] A = {{1, 0, 0},  {0, 1, 0},  {0, 0, 1}}; | Identity Matrix |

**Task 10**

You're tasked with creating a **"Treasure Hunt"** game, where a player navigates a 2D grid to find hidden treasure. In this grid:

* The number 7 represents the player’s current position.
* The number 10 represents the treasure.
* The number -1 represents mines that end the game if stepped on.
* The number 0 represents open spaces.

The player begins with 5 moves to reach the treasure. Moving outside the grid or onto a mine will end the game. Even failing to reach the treasure within 5 moves will result in a loss. The player can only move straight (UP / DOWN / LEFT / RIGHT).

You have given a skeleton code for this problem. Complete the code to solve the problem. [[Link to code](https://drive.google.com/file/d/12CI2nNLhC_0u7w5_hYK7aBn9o_NzAQrl/view?usp=drive_link)]

Note: Initial grid can be changed. So solve accordingly.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| Enter move 1: RIGHT  Enter move 2: UP  Enter move 3: UP  Enter move 4: LEFT  Enter move 5: UP | Initial Map:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 7 0 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 0 7 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 7 0  0 -1 0 0 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 0 7 -1  -1 0 -1 0 0  0 -1 0 0 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 7 0 -1  -1 0 -1 0 0  0 -1 0 0 -1  0 -1 0 -1 0  Treasure found. You win!  Final state:  0 0 7 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 0 0 -1  0 -1 0 -1 0 |
| Enter move 1: RIGHT  Enter move 2: UP  Enter move 3: LEFT | Initial Map:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 7 0 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 0 7 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 7 0  0 -1 0 0 -1  0 -1 0 -1 0  Player stepped on mine. Game Over! |
| Enter move 1: DOWN  Enter move 1: DOWN | Initial Map:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 7 0 -1  0 -1 0 -1 0  Current state:  0 0 10 0 -1  0 -1 0 0 -1  -1 0 -1 0 0  0 -1 0 0 -1  0 -1 7 -1 0  Player fell outside the playing area. Game over! |

For this course, we’ll be using **DrJava** as IDE for Java Coding:

[**Link to JDK and DrJava**](https://drive.google.com/drive/folders/1pzTIouwS9-tHVEb9HdBnp1WL7NDrK_BI)

**Drjava Installation Guide:**

[**https://www.youtube.com/watch?v=Gss9sL3Q-8s**](https://www.youtube.com/watch?v=Gss9sL3Q-8s)

**Ungraded Tasks (Optional)**

(You don’t have to submit the ungraded tasks)

**Task 1**

Write a Java program that will take an integer number N from the user and create an integer array by taking N numbers from the user. Then take another number from the user and create a new array by removing that number from the input array. Finally, print the new array.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| N = 5  23  100  0  56  -34  Remove Element = 100 | Input array:  23 100 0 56 -34  New array:  23 0 56 -34 |
| N = 4  -5  10  2  -7  Remove Element = 43 | Input array:  -5 10 2 -7  Element not found |

**Task 2**

Write a program that reads 5 numbers into an array and prints the smallest and largest number and their location in the array.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 7  13  2  10  6 | The largest number 13 was found at location 1.  The smallest number 2 was found at location 2. |
| 2  4  -5  12  3 | The largest number 12 was found at location 3.  The smallest number -5 was found at location 2. |

**Task 3**

Write a program that asks the user how many numbers to take. Then, it takes that many numbers in an array and prints the median value.

[How to Find the Median Value: <http://www.mathsisfun.com/median.html>]

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 5  10  50  40  20  30 | The median is 30.  **Explanation:** 30 falls in middle 10, 20, 30, 40, 50 |
| 4  30  10  40  20 | The median is 25.  **Explanation:** (20+30)/2=25 (average of two middle values from 10, 20, 30, 40. |

**Task 4**

You are given a matrix **A** of size M×N. Write a Java program that will take an integer number **k** from the user and perform scalar multiplication A = **k\*A**

|  |  |
| --- | --- |
| **Given Array** | **Output** |
| int [ ] [ ] A = {{4, 5, 7},  {0, 3, -2},  {4, 1, -4},  {5, 10, 1}};  k = 4 | 16 20 28  0 12 -8  16 4 -16  20 40 4 |
| int [ ] [ ] A = {{1, 2, 4},  {5, 7, 2}};  k = 3 | 3 6 12  15 21 6 |

**Task 5**

Write a Java program that will take M and N from the user and create a matrix A of dimension M×N*.*  Print the matrix A. Then you have to transpose the matrix in a new 2D array. Finally, print the new array.

*The transpose of a matrix is a new matrix that is obtained by exchanging the rows and columns of the original matrix. Given a matrix A with dimensions M×N, the transpose AT will have dimensions N×M, where the rows of A become the columns of AT and vice versa.*

|  |  |
| --- | --- |
| **Given Array** | **Output** |
| M = 3  N = 3  1  2  3  4  5  6  7  8  9 | Matrix A  1 2 3  4 5 6  7 8 9  Transpose A  1 4 7  2 5 8  3 6 9 |
| M = 2  N = 4  11  2  3  4  1  4  9  16 | Matrix A  11 2 3 4  1 4 9 16  Transpose A  11 1  2 4  3 9  4 16 |