****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Summer,Year:2025),B.Sc.in CSE (Day)**

**LAB REPORT NO - 02**

**Course Title: Operating System Lab**

**Course Code: CSE 310 Section:** **223-D2**

**Lab Experiment Name :Lab Manuals -04 Lab Exercise**

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | **MD.SHAJALAL** | **223002088** |

**Submission Date : 09 – 08 - 2025**

**Course Teacher’s Name : Umme Habiba**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

**Objective :**

The objective of this lab is to gain practical knowledge of **shell scripting** by implementing various programs that use loops, arrays, functions, and string processing.  
Specifically, the experiments aim to:

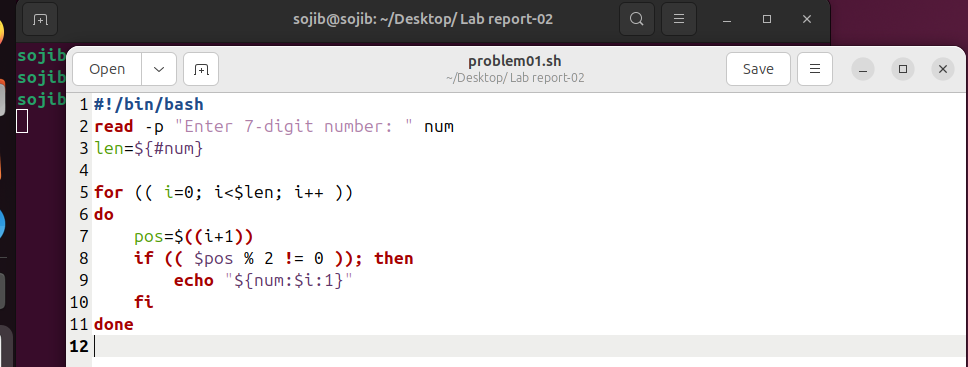
1. Extract digits from specific positions in a number using a for loop.
2. Count occurrences of each digit using a while loop and associative arrays.
3. Identify the 2nd and 3rd largest numbers from a list and compute their sum using array manipulation.
4. Calculate the factorial of two numbers using user-defined functions and determine their sum.
5. Count the total number of alphabets, digits, and special characters in a string using pattern matching.

**Q.1. Write a shell program to display odd position numbers (using For loop).**

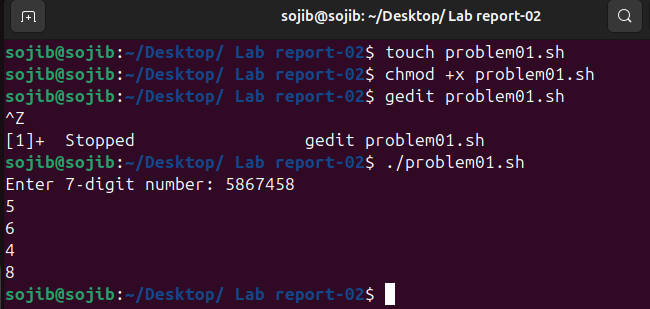
**Algorithm:**

1. Read a 7-digit number from the user.
2. Calculate the length of the number.
3. Use a for loop to iterate over each digit.
4. Check if the position is odd.
5. Print digits at odd positions.

Code:



Sample Output:

****

**Q.2. Write a Shell program using while loop:**

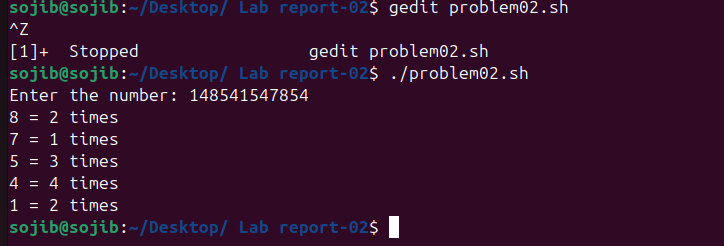
**Algorithm:**

1. Read a number as a string.
2. Initialize an associative array to store counts.
3. Use a while loop to process each character.
4. Increment the count for each digit found.
5. Display the counts.

**Code:**



Sample Output:

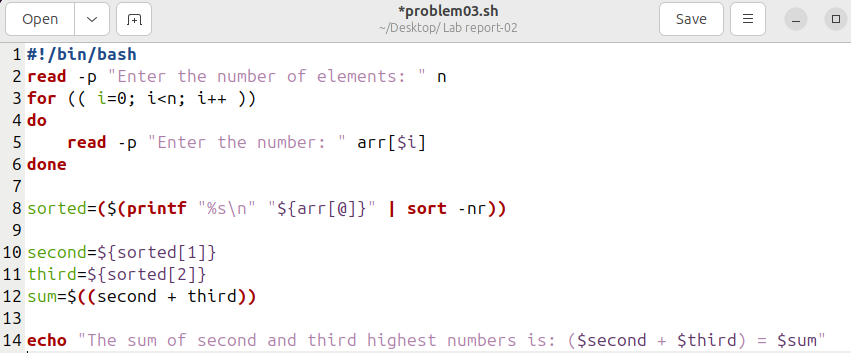
****

**Q.3.** **Write a Shell program to find the 2nd highest and 3rd highest numbers from a set of numbers and sum of them using array.**

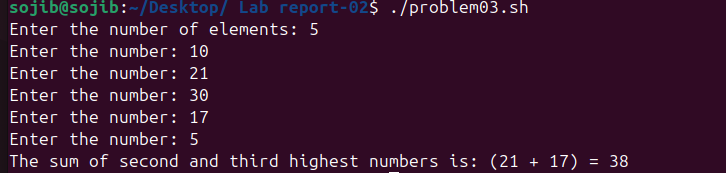
**Algorithm:**

1. Read n numbers from the user.
2. Store them in an array.
3. Sort the array in descending order.
4. Extract the 2nd and 3rd largest numbers.
5. Calculate and display their sum.

**Code:**



Sample Output:

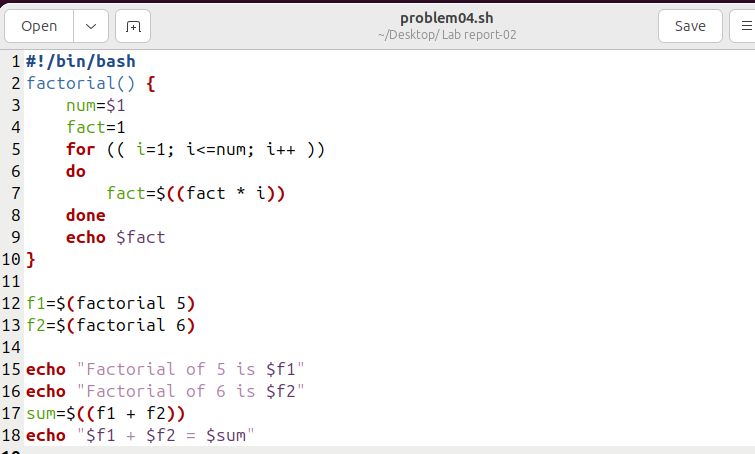


**Q.4.** **Write a Shell program to find the factorial of two different numbers and sum of the numbers using function.**

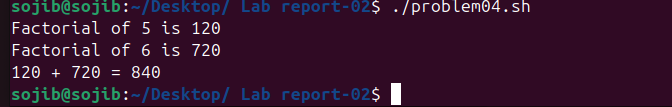
**Algorithm:**

1. Define a function factorial to calculate factorial.
2. Call the function for each number.
3. Add the results.
4. Display the factorials and the sum.

**Code:**



Sample Output:

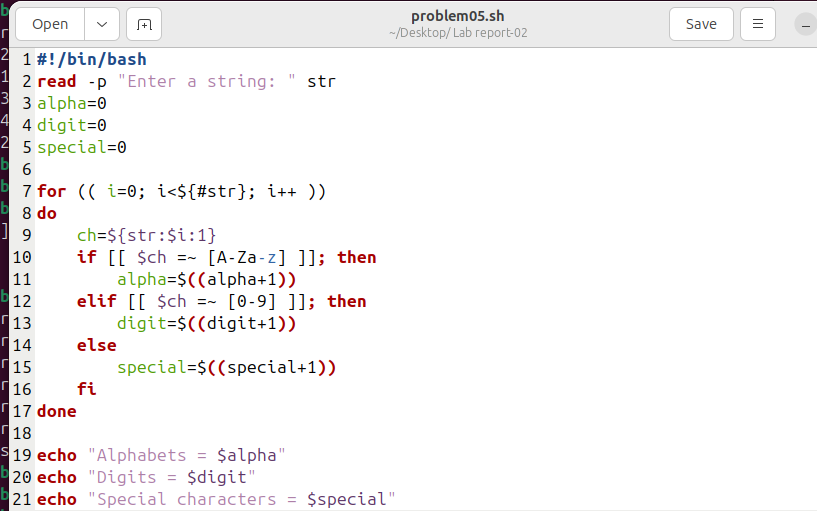


**Q.5. Write a Shell program to find total number of alphabets, digits or special characters in a string.**

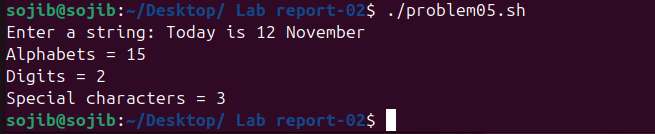
**Algorithm:**

1. Read the string from the user.
2. Initialize counters for alphabets, digits, and special characters.
3. Loop through each character and classify it.
4. Display the counts.

**Code:**



Sample Output:



**Conclusion :**

From these experiments, it can be concluded that **shell scripting** is a powerful and flexible tool for automating tasks and processing data in UNIX/Linux environments.  
The key takeaways include:

* **Looping constructs** (for and while) are essential for iterating over strings, numbers, and arrays.
* **String indexing** in bash allows easy extraction of specific characters or substrings.
* **Associative arrays** can be used to efficiently store and retrieve frequency counts.
* **Sorting techniques** combined with arrays enable quick selection of maximum or specific-order elements.
* **Functions** promote code reusability and make programs easier to maintain.
* **Pattern matching with regular expressions** helps classify characters in strings accurately.