Experiment-4

Student Name: Sahil Ittan **UID:** 22BCS14503

Branch: BE-CSE **Section/Group:** KPIT-902/B

Semester: 6th **Date of Performance:**15/01/25

Subject Name: Adv. Prog. Lab - 2 **Subject Code:** 22CSP-351

1. **Aim:**

   Problem: 1.4.1: Given two strings and goal, return true if and only if s can become goal after some number of shifts on s. A shift on s consists of moving the leftmost character of s to the rightmost position.

   Problem: 1.4.2: Given an array nums containing n distinct numbers in the range [0, n], return the only number in the range that is missing from the array.

2. **Objective:**

   1. **Rotate String.**
      - To practice fundamental string manipulation techniques, such as slicing, rearranging, and concatenating substrings.
      - To develop the logic for shifting string characters and exploring how rearrangements can be used to check equivalence or create new strings.
      - To introduce error handling for invalid inputs, such as strings that are too short for meaningful operations.
   2. **Missing Number.**
      - To practice basic array operations such as sorting and traversal.
      - To implement a logical approach for identifying a missing element by comparing indices with expected values.
      - To handle edge cases where the missing number could be at the beginning, middle, or end of the range.

3. **Implementation/Code:**

1.)

```
#include <iostream> #include <string>
using namespace std;
string rearrangeString(const string& input)
{
int n = input.size(); if (n < 5)
```

```
{
return "String too short to rearrange";
}
string lastThree = input.substr(n - 3);
string firstTwo = input.substr(0, 3);
return lastThree + firstTwo;
}
int main()
{
string s;
cout << "Enter a string: "; cin >> s;
string result = rearrangeString(s); cout<<"Rearranged string: "<<result<<endl;
cout<<"Name: Sahil Ittan"<<endl; cout<<"Uid: 22BCS14503"<<endl;
return 0;
}
```

Output 1:

```
Enter a string: SahilIttan
Rearranged string: tanSah
Name: Sahil Ittan
Uid: 22BCS14503
```

2.)

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;
int findMissingElement(const vector<int>& arr)
{
vector<int> sortedArr = arr;
sort(sortedArr.begin(),
sortedArr.end());

        for (int i = 0; i <
        sortedArr.size();++i)
        {if (sortedArr[i] != i) {
```

```
        return i;
                }}
        return sortedArr.size();
    }
    int main()
    {
        int n;
        cout<<"Enter the number of elements in the array: ";
        cin>>n;

    vector<int> arr(n);
        cout<<"Enter the elements of the
    array: ";
        for (int i=0; i<n; ++i)
        {
                cin>>arr[i];
        }
int missingElement = findMissingElement(arr);
cout<<"The missing element is: "<<missingElement<<endl;
cout<<"Name: Sahil Ittan"<<endl;
cout<<"Uid: 22BCS14503"<<endl;
return 0;
    }
```

Output 2:

```
Enter the number of elements in the array: 10
Enter the elements of the array: 0
1
2
3
4
5
6
8
9
10
The missing element is: 7
Name: Sahil Ittan
```

4. **Time Complexity:**
   1. O(n)
   2. O(n)
5. **Space Complexity:**
   1. O(n)
   2. O(1)
6. **Learning Outcome:**
   1. Understand how to slice substrings using substr and rearrange them by concatenating different parts.
   2. Learn to handle edge cases, such as ensuring the string length meets minimum requirements before performing operations.
   3. Explore how to simulate shifts or rotations in a string for further logical checks or transformations.
   4. Learn to compare array indices with their expected values in a sorted array to identify discrepancies.
   5. Develop an understanding of how to solve problems involving missing elements using structured, step-by-step approaches.Practical Problem-Solving: Enhance practical problem-solving skills applicable to real-world scenarios involving array manipulation and optimization.