Ahsania Mission University of Science & Technology

Department of Computer Science and Engineering

1st Batch, 2nd Year 2st Semester, Fall 2025

Course Code: CSE 2201

Course Title: Computer Algorithms

Lab Report

Experiment No :01

Experiment Date :05/05/2025

Submission date:07/05/2025

Submitted To

Md.Fahim Faisal,Lecturer.
Department of Computer Science and Engineering
Faculty of Engineering, Ahsania Mission University of Science & Technology

Submitted By

Name : **MD:Rubyait Roman Rifat**

ID : **1012320005101014**

Task 1:

You are given a template code in the IDE.

Update the code to merge the 2 arrays based on the process defined above.

SOURCE CODE:

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    int arr1[100] = {2, 4, 6}; // First array
    int size1 = 3;
    int arr2[100] = {8, 10, 12, 14}; // Second array
    int size2 = 4;
    // Merged array
    int merged[200];
    int mergedSize = size1 + size2;
    // Copy elements from arr1 to merged
    for (int i = 0; i < size1; i++) {
        merged[i] = arr1[i];
    }
    // Copy elements from arr2 to merged
    for (int i = 0; i < size2; i++) {
        merged[size1 + i] = arr2[i];
    }
    // Print the merged array
    for (int i = 0; i < mergedSize; i++) {
        cout << merged[i] << " ";
    }
```

```
    return 0;
}
```

OUTPUT:

2 4 6 8 10 12 14

Process returned 0 (0x0)   execution time : 0.054 s

Press any key to continue.

***Problem: Sum of Array elements

Given an array A, Output the sum of all elements in A.

Input Format

• The first line of input will contain a single integer N denoting the number of elements in A.

• the second line contains N space-separated integers denoting elements of the array A.

Output Format

Output a single integer, sum of all the elements in the array A.

Sample 1:

Input

8 2 4 1 4

19

Output

5

SOURCE CODE:

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    int N;
    cin >> N; // Input the number of elements
```

```
  int A[N];
   for (int i = 0; i < N; i++) {
       cin >> A[i]; // Input each element
   }
   // Calculate the sum using accumulate
   int sum = accumulate(A, A + N, 0);
   cout << sum << endl; // Output the sum
   return 0;
}
```

OUTPUT:

```
5
8 2 4 1 4
19


Process returned 0 (0x0)   execution time : 53.461 s
Press any key to continue.
```

Task No: 02

Problem: Find maximum in an Array

Given a list of N integers, representing height of mountains. Find the height of the tallest mountain.

Input:

• First line will contain T, number of testcases. Then the testcases follow.

• The first line in each testcase contains one integer, N.

• The following line contains N space separated integers: the height of each mountains.

Output:

For each testcase, output one line with one integer: the height of the tallest mountain for that test case.

Constraints

• 1≤ T≤10

• 1 ≤N≤100000

• 0≤height of each mountain ≤109

SOURCE CODE:

```
#include <bits/stdc++.h>

using namespace std;

int main() {
   int T;
   cin >> T;  // Number of test cases
   while (T--) {
      int N;
      cin >> N;  // Number of mountains
      int maxHeight = 0;
      for (int i = 0; i < N; ++i) {
         int height;
         cin >> height;
         if (height > maxHeight) {
            maxHeight = height;
         }
      }
      cout << maxHeight << endl;
   }
   return 0;
```

}

OUTPUT:

2

5

1 3 2 5 4

5

3

10 20 15

20

Process returned 0 (0x0) execution time : 0.976 s

Press any key to continue.

Task: 03

Problem: MIN To MAX

You are given an array A of size N.

Let M be the minimum value present in the array initially.

In one operation, you can choose an element Ai ($1 \le i \le N$) and an integer X ($1 \le X \le 100$), and set Ai = X.

Determine the minimum number of operations required to make M the maximum value in the array A.

Input Format

• The first line of input will contain a single integer T, denoting the number of test cases.

• Each test case consists of multiple lines of input:

o The first line of each test case contains a single integer N - the size of the array.

o The next line of each test case contains N space-separated integers A1, A2, ..., An - the

elements of the array.

Output Format

For each test case, output on a new line, the minimum number of operations required to make M the

maximum value in the array A.

Constraints

• $1 \leq T \leq 100$

• $1 \leq N \leq 100$

• $1 \leq Ai \leq 100$

Sample 1

Input:

3

2

1 2

4

2 2 3 4

1

1

Output:

1

2

0

Explanation:

Test case 1:

The minimum value in the array, M, is initially 1. We can use one operation as follows:

• Choose A2 and set it as X = 1. Thus, the final array becomes [1, 1].

• Since all elements of the final array are 1, the maximum value of the array is now 1.

• It can be shown that this is the minimum number of operations required.

Test case 2:

The minimum value in the array, M, is initially 2. We can use two operations as follows:

• Choose A4 and set it as X = 2. The array becomes [2, 2, 3, 2].

• Choose A3 and set it as X = 2. The array becomes [2, 2, 2, 2].

• Since all elements of the final array are 2, the maximum value of the array is now 2.

Test case 3:

The minimum value in the array, M, is initially 1. It is also the maximum value of the array. Hence, no

operations are required.

SOURCE CODE:

```
#include <bits/stdc++.h>

using namespace std;

int main() {
   int T;
   cin >> T; // Number of test cases
   while (T--) {
      int N;
      cin >> N; // Size of the array
      vector<int> A(N);
```

```
        for (int i = 0; i < N; ++i) {
            cin >> A[i]; // Input array elements
        }
        int M = *min_element(A.begin(), A.end()); // Find initial minimum value
        int operations = 0;
        for (int i = 0; i < N; ++i) {
            if (A[i] > M) {
                ++operations; // Count elements greater than M
            }
        }
        cout << operations << endl; // Output the result
    }
    return 0;
}
```

OUTPUT:

```
3
2
1 2
1
4
2 2 3 4
2
1
1
0
```

Process returned 0 (0x0)   execution time : 0.951 s

Press any key to continue.

Task No: 04

Pseudocode: Grade School Integer Multiplication

Explanation

• We multiply each digit of B by every digit of A, like in long multiplication.

• Each partial result is adjusted for its place value and added to the final result.

• Carry is handled at every step to ensure correct arithmetic.

• The final sum gives the correct product.

SOURCE CODE:

```
#include <iostream>
#include <cstring> // For memset
using namespace std;
#define MAX 200 // Maximum digits to handle large numbers
class BigIntMultiplication {
private:
   int numA[MAX], numB[MAX], result[MAX];
   int lenA, lenB;
public:
   // Constructor to initialize arrays
   BigIntMultiplication() {
      memset(numA, 0, sizeof(numA));
      memset(numB, 0, sizeof(numB));
      memset(result, 0, sizeof(result));
      lenA = lenB = 0;
```

```
}
// Function to convert an integer into a digit array (least significant digit first)
void storeNumber(int num, int arr[], int &length) {
    while (num > 0) {
        arr[length++] = num % 10; // store digit
        num /= 10;
    }
}
// Function to multiply two integers using grade school multiplication
void multiply(int A, int B) {
    if (A == 0 || B == 0) {
        cout << "0" << endl; // If either number is 0, the product is 0
        return;
    }
    // Store numbers in digit arrays
    storeNumber(A, numA, lenA);
    storeNumber(B, numB, lenB);
    // Perform grade school multiplication
    for (int i = 0; i < lenB; i++) {
        int carry = 0;
        for (int j = 0; j < lenA; j++) {
            int temp = numB[i] * numA[j] + result[i + j] + carry;
            result[i + j] = temp % 10;
            carry = temp / 10;
        }
        if (carry > 0) {
```

```
                result[i + lenA] += carry;
            }
        }
        printResult();
    }
    // Function to print the final result
    void printResult() {
        int lenResult = lenA + lenB;
        while (lenResult > 1 && result[lenResult - 1] == 0) {
            lenResult--; // Remove leading zeros
        }
        // Print the result in correct order (reverse of storage)
        for (int i = lenResult - 1; i >= 0; i--) {
            cout << result[i];
        }
        cout << endl;
    }
};
// Driver Code
int main() {
    int A, B;
    cout << "Enter two integers: ";
    cin >> A >> B;
    BigIntMultiplication multiplier;
    cout << "Product: ";
    multiplier.multiply(A, B);
```

```
    return 0;
}
```

OUTPUT:

```
Enter the first number: 98765
Enter the second number: 43210
Result: 4267635650
Process returned 0 (0x0)   execution time : 18.543 s
Press any key to continue.
```