Ahsania Mission University of Science & Technology

Lab Report

Lab No: 01

Course Code: CSE 2202

Course Title: Computer Algorithm Sessional.

Submitted By:

MT.Sumaiya Ajmeri Borsha
ID: 1012320005101020
1st Batch, 2nd Year, 2nd Semester
Department of Computer science and Engineering,
Ahsania Mission University of Science & Technology

Submitted To:

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

Task No.: 01

Problem Statement: Merging 2 arrays Merging two 1D arrays involves combining the elements of both arrays into a single array while maintaining their original order. Here's a step-by-step process for merging two 1D arrays:

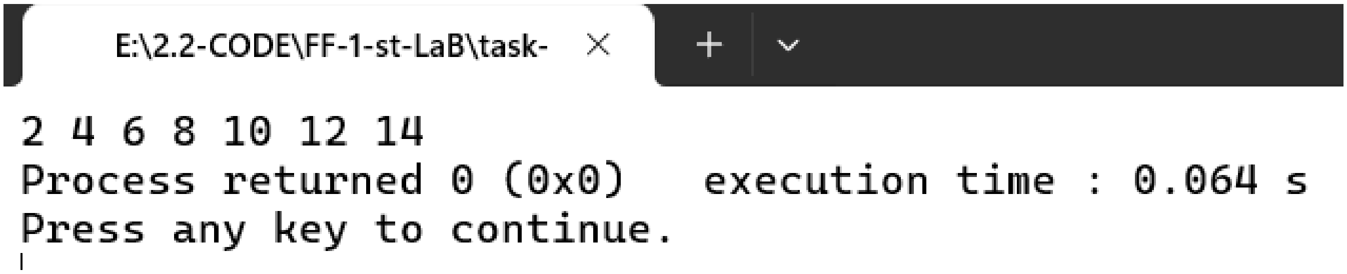
- Create a New Array: Create a new array that is large enough to hold the combined elements of both arrays. The size of the new array should be the sum of the sizes of the two original arrays.
- Copy Elements: Iterate through the elements of the first array and copy them to the new array. Then, iterate through the elements of the second array and copy them to the new array after the elements of the first array.
- Result: The new array now contains all the elements from both original arrays, merged in the desired order.

Source Code:

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

using namespace std;

int main()

{

   int arr1[100]={2, 4, 6};

   int size1=3;

   int arr2[100]={8, 10, 12, 14};

   int size2=4;

   int merged_size=size1+size2;

   int arr3[merged_size];

   for(int i=0;i<size1;i++)

   {

      arr3[i]=arr1[i];

      if(i==2)

      {

         for(int j=0;j<size2;j++)
```

```
            {
                i++;
                arr3[i]=arr2[j];
            }
        }
    }
    for(int i=0;i<merged_size; i++)
    {
        cout<<arr3[i]<<" ";
    }
}
```

Output:



```
2 4 6 8 10 12 14
Process returned 0 (0x0)   execution time : 0.064 s
Press any key to continue.
```

Task No.: 02

Problem Statement: 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

5

8 2 4 1 4

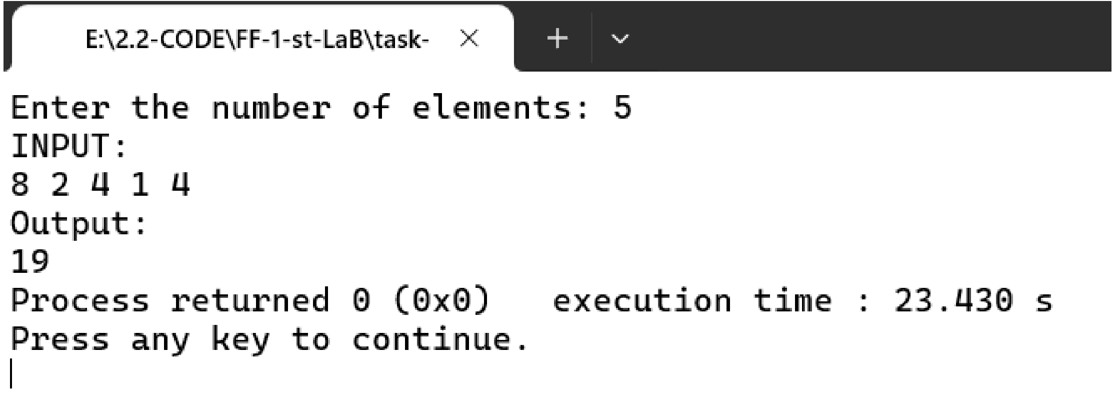
Output

19

Source Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   int n,sum=0;
   cout<<"Enter the number of elements: ";
   cin>>n;
   int arr[n];
   cout<<"INPUT:" <<endl;
   for(int i=0; i<n; i++)
   {
      cin>>arr[i];
      sum=sum+arr[i];
   }
   cout<<"Output: "<<endl;
   cout<<sum;
   return 0;}
```

Output:



```
Enter the number of elements: 5
INPUT:
8 2 4 1 4
Output:
19
Process returned 0 (0x0)   execution time : 23.430 s
Press any key to continue.
```

Task No.: 03

Problem Statement: 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

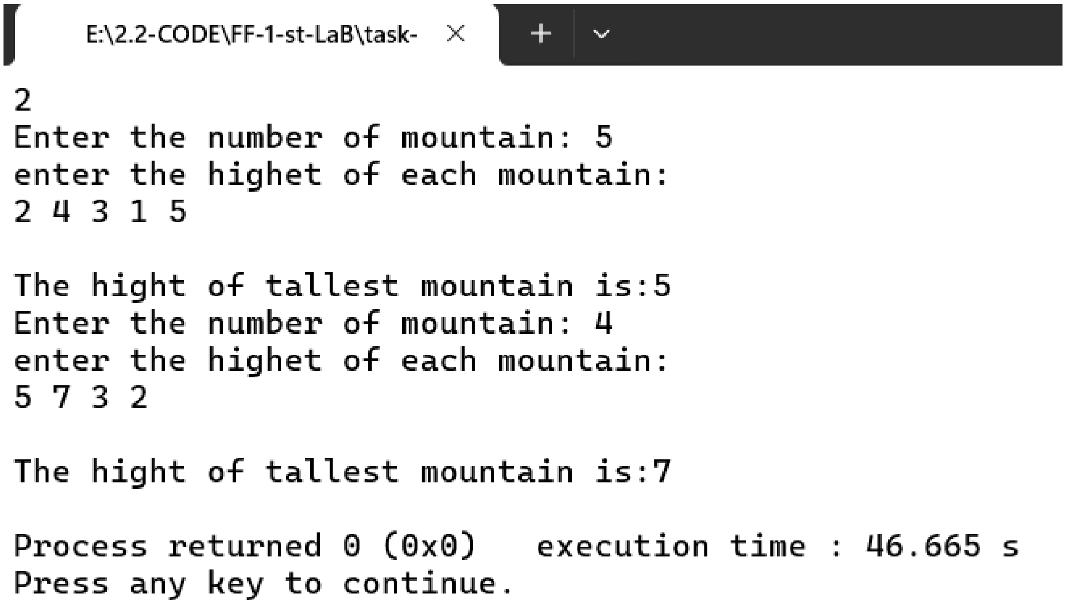
- 1≤ T≤10
- 1 ≤N≤100000
- 0≤height of each mountain ≤109

Source Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   int T;
   cin>>T;
   while(T--)
```

```
  {
    int n,Max=0;
    cout<<"Enter the number of mountain: ";
    cin>>n;
    int arr[n];
    cout<<"enter the highet of each mountain:"<<endl;
    for(int i=0; i<n; i++)
    {
      cin>>arr[i];
      if(arr[i]>Max)
      {
        Max=arr[i];
      }
    }
    cout<<"\nThe hight of tallest mountain is:" ;
    cout<<Max<<endl;
  }
}
```

Output:



```
2
Enter the number of mountain: 5
enter the highet of each mountain:
2 4 3 1 5

The hight of tallest mountain is:5
Enter the number of mountain: 4
enter the highet of each mountain:
5 7 3 2

The hight of tallest mountain is:7

Process returned 0 (0x0)   execution time : 46.665 s
Press any key to continue.
```

Task No.: 04

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

- $1 \le T \le 100$
- $1 \le N \le 100$
- $1 \le Ai \le 100$

Sample 1

Input:

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

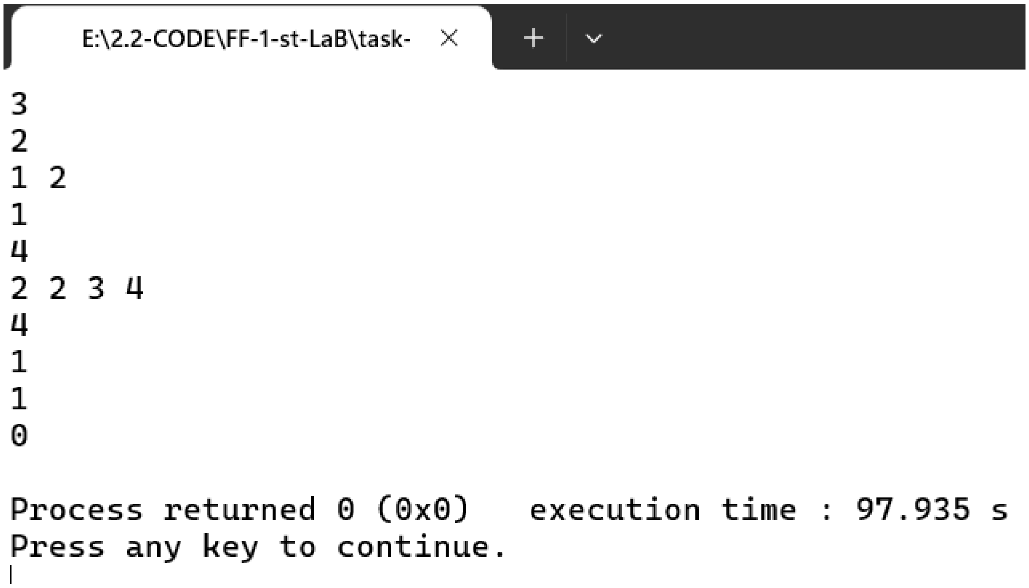
Output:

```
1
2
0
```

Source Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int T,M=100;
    cin>>T;
    while(T--)
    {
        int n;
        cin>>n;
        int arr[n];
        for(int i=0; i<n; i++)
        {
            cin>>arr[i];
        }
        for(int i=0; i<n; i++)
        {
            if(M>arr[i])
            {
                M=arr[i];
            }
        }
        int operations = 0;
        for (int i = 0; i < n; i++)
        {
            if (arr[i] > M)
            {
                operations++;
            }
  }
        cout << operations << endl;
    }
}
```

Output:



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

Process returned 0 (0x0)   execution time : 97.935 s
Press any key to continue.
```

Task No.: 05

Problem Statement: Grade School Integer Multiplication

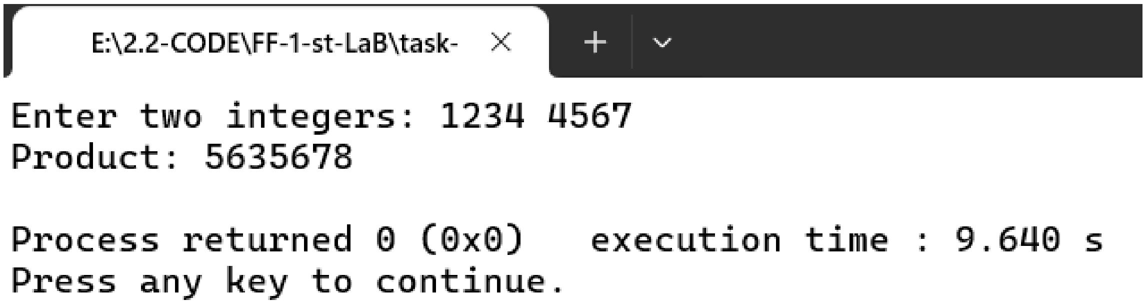
Source Code:

```
#include <iostream>
#include <cstring>
using namespace std;
#define MAX 200

class BigIntMultiplication
{
private:
   int numA[MAX], numB[MAX], result[MAX];
   int lenA, lenB;
public:
   BigIntMultiplication()
   {
      memset(numA, 0, sizeof(numA));
      memset(numB, 0, sizeof(numB));
      memset(result, 0, sizeof(result));
      lenA = lenB = 0;
```

```
}
void storeNumber(int num, int arr[], int &length)
{
    while (num > 0)
    {
        arr[length++] = num % 10;
        num /= 10;
    }
}
void multiply(int A, int B)
{
    if (A == 0 || B == 0)
    {
        cout << "0" << endl;
        return;
    }
    storeNumber(A, numA, lenA);
    storeNumber(B, numB, lenB);
    for (int i = 0; i < lenA; i++)
    {
        for (int j = 0; j < lenB; j++)
        {
            result[i + j] += numA[i] * numB[j];
            result[i + j + 1] += result[i + j] / 10;
            result[i + j] %= 10;
        }
    }
    printResult();
}

void printResult()
{
    int lenResult = lenA + lenB;
    while (lenResult > 1 && result[lenResult - 1] == 0)
    {
        lenResult--;
    }
    for (int i = lenResult - 1; i >= 0; i--)
    {
        cout << result[i];
    }
```

```
        cout << endl;
    }
};

int main()
{
    int A, B;
    cout << "Enter two integers: ";
    cin >> A >> B;
    BigIntMultiplication multiplier;
    cout << "Product: ";
    multiplier.multiply(A, B);
    return 0;
}
```

Output:



```
Enter two integers: 1234 4567
Product: 5635678

Process returned 0 (0x0)   execution time : 9.640 s
Press any key to continue.
```