**Experiment No. 1**

**Title: Interpretation of problem statement and Identification of test cases for given problem statement**

**Batch: B2 Roll No: 16010421119 Experiment No.:1**

### Aim: To interpret given problem statement and identify test cases for given problem statement

**Resources needed:** Text Editor, C/C++ IDE

### Theory:

### In competitive programming the problem statement is mostly given pertaining to real-world scenario. Along with input and output information, constraints on input/output are also given most of the times with the problem statement. Hence in competitive programming to get started with a problem, the first step is to read and understand the problem statement and given information and find the following details from it:

### Identify the input values

### Identify the constraints on input

### Identify the output values

### Identify the constraints on output

### Along with problem statement and constraints information, input format and output format information is also given. To get a clear understanding of these formats, sample input values and it’s corresponding output values are also given. We refer to these values as sample test cases.

### A Test Case is some sample input value and it’s expected out value. In competitive programming, with every problem statement, some sample test cases are given most of the times. But these sample test cases do not cover all the general and special cases. Since the competitive programming platforms evaluate the solution based on test cases, it is essential to identify the general and special test cases for the problem statement under consideration. Special Cases mostly require special handling in the solution.

### Test Cases can be identified using following approach:

### Random Value Test Cases – Here some random values of input and it’s corresponding output within the given input/output constraints can be considered

### Minimal Value Test Cases – Here considering factors, such as minimum number of total input values or minimum possible value for a input, special cases can be identified

### Maximal Value Test Cases - Here considering factors, such as maximum number of total input values or maximum possible value for a input, special cases can be identified. These test cases also help to determine problems like integer overflow, crash point of solution, maximum running time and memory requirement of the solution and so on.

### Activity:

Consider the following problem statement and other information provided along with it:

**Problem**

You are provided an array of size that contains non-negative integers. Your task is to determine whether the number that is formed by selecting the last digit of all the N numbers is divisible by 10.

**Input format**

* First line: A single integer N denoting the size of array A
* Second line: N space-separated integers.

**Output format**

If the number is divisible by 10, then print ‘*Yes’*. Otherwise, print ‘*No’*

**Constraints**1 ≤ N ≤ 105

0 ≤ A[i] ≤ 105

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 5  45 23 65 22 74 | No |

**Task 1:**

Identify the following from the given information:

1. Input values
2. Constraints on input values
3. Output values
4. Constraints on output values
5. Specified format for input values
6. Specified format for output values

**Task 2:**

Identify general and special test cases for given problem statement. List down in all 10 - 12 test cases in table format as shown:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Sample Input** | **Sample Output** | **Description** | **Test Case Type (general/special)** |
| 1. | 5  45 23 65 22 74 | No | array with 5 integer numbers | general |
| 2. | -2  10 20 | Segmentation Fault(Negative Size is not possible) | Negative Size of an array | Special Case |
| 3. | 5  10 20 30 40 50 | Yes | .An array of size 5 | General case…. |
| 4. | 1000000 | Constraint violated  1 ≤ N ≤ 105 | An array exceeding the limitations of Size. | Special Case |
| 5. | 7  10000000 2343546 364646 26164274 3252316 32616 2315246 | Constraint Violated  0 ≤ A[i] ≤ 105 | An array exceeding the limitation of element. | Special Case |
| 6. | 7  -3 4 6 6 -6 35 -24 | Constraint violated-- non-negative integers | An array containing negative elements | Special Case |
| 7. | 10  6 7 8 423 436 326 574 57 78 35 | Size cannot exceeding limitation of ‘int’ | An array where user input for size is a string | Special Case |
| 8. | 0 | Your code didn't print anything. | An array having size 0 | Special Case |
| 9. | 1  0 | Yes | Minimum Test Case | General case |
| 10. | 7  100000 100000 100000 100000 100000 100000 100000 | Yes | Maximum Test Case | General case |

### Solution:

### #include <iostream>

### using namespace std;

### int main()

### {

### int n;

### cin>>n;

### int arr[n];

### if(n>=1)

### {

### for(int i=0;i<n;i++)

### {

### 

### cin>>arr[i];

### 

### 

### }

### if((arr[n-1]%10)==0)

### {

### cout<<"Yes";

### }

### else

### {

### cout<<"No";

### }

### }

### 

### return 0;

### }

### ClassWork –

### 

### 

### 

### Outcomes:

### CO1. Inculcate the best practices that are essential for competitive programming

### Post Lab Questions:

### Consider the given problem statement and related information:

### Problem

You have been given a positive integer N where 1 ≤ N ≤ 12. You need to find and print the Factorial of this number.

### Input Format

### The first and only line of the input contains a single integer N denoting the number whose factorial you need to find.

### #include <iostream>

### using namespace std;

### int factorial(int n)

### {

### int fac = 1;

### for(int i = 1;i<=n;i++)

### {

### fac = fac \* i;

### }

### return fac;

### }

### int main() {

### int fac;

### cout<<"Enter the number you want the factorial of: ";

### cin>>fac;

### if(fac>12)

### {

### cout<<"This violates input constraint!!";

### }

### else if(fac==0)

### {

### cout<<"Factorial is: "<<1;

### }

### else

### {

### int x = factorial(fac);

### cout<<"Factorial is: "<<x;

### }

### 

### return 0;

### }

### Output Format

### Output a single line denoting the factorial of the number N.

### 

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 3 | 6 |

**Task 1:**

Identify the following from the given information:

1. Input values
2. Constraints on input values
3. Output values
4. Constraints on output values
5. Specified format for input values
6. Specified format for output values

Input Values – 1-12

Input Constraints -- 1<= N <= 12

Output values – Positive integer

Constraints on output values – Positive integer

Specified format for input values – Positive Integer

Specified format for output values – Positive Integer

**Task 2:**

Identify general and special test cases for given problem statement. List down in all 6-8 test cases in table format (refer activity section for table format of test cases)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Sample Input** | **Sample Output** | **Description** | **Test Case Type (general/special)** |
| 1. | 1 | 1 | Minimum Test Case | General |
| 2. | 12 | 479001600 | Maximum Test Case | General |
| 3. | 0 | 1 | Factorial of 0 is 1 | Special Case |
| 4. | 13 | This violates the constraint | An element exceeding the limitations of N. | Special Case |
| 5. | -3 | This violates the constraint | Negative integer factorial is not allowed | Special Case |
| 6. | 4 | 24 | Normal Case | Special Case |

**Conclusion: (Conclusion to be based on the objectives and outcomes achieved)**

**We can conclude that we have learnt about the essential techniques for competitive programming..**

**References:**

1. Antti Laaksonen, “Guide to Competitive Programming”,Springer,2018
2. Gayle Laakmann McDowell,” Cracking the Coding Interview”,CareerCup LLC,2015
3. Steven S. Skiena Miguel A. Revilla,”Programming challenges, The Programming Contest Training Manual”, Springer, 2006
4. Antti Laaksonen, “Competitive Programmer’s Handbook”, Hand book, 2018
5. Steven Halim and Felix Halim, “Competitive Programming 3: The Lower Bounds of Programming Contests”, Handbook for ACM ICPC