## Experiment No. 1

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

**Batch:B1** **Roll No: 16010420133** **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:

1. Identify the input values
2. Identify the constraints on input
3. Identify the output values
4. 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:

1. Random Value Test Cases – Here some random values of input and it's corresponding output within the given input/output constraints can be considered
2. 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
3. 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**
$1 \leq N \leq 10^5$
$0 \leq A[i] \leq 10^5$

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

**Task 1:**
Identify the following from the given information:
1. Input values
    - Input value will be the two integers.
    - First integer will be the size of the number of elements user will enter
    - Second Integer will be the n- number of elements

2. Constraints on input values
    - Here, Input value should be positive integer
    - It should be equal to the size of array assigned by user.
    - It should not be a negative number
    - Not a floating number.
    - Number should be in range 0 to 32767.
    - It should be separated by space.
    - It shouldn't be a character or a special character.
    - After entering the last element user should press the Enter.

3. Output values

Output values will be:

- "Yes" or
- "No"

Depending on the input values

4. Constraints on output values
    - Output value should return single answer on the basis on input values means either "Yes" or "No"

5. Specified format for input values
    - Input values should be separated by space and be a positive integer.
    - It should be in a specified range.
    - User must press enter after entering the last element.

6. Specified format for output values
    - Output value should only return yes or no. It cannot return both at the same time.
    - If it is divisible by 10 then it should return yes.
    - If it is not divisible by 10 then it should return No.

**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:

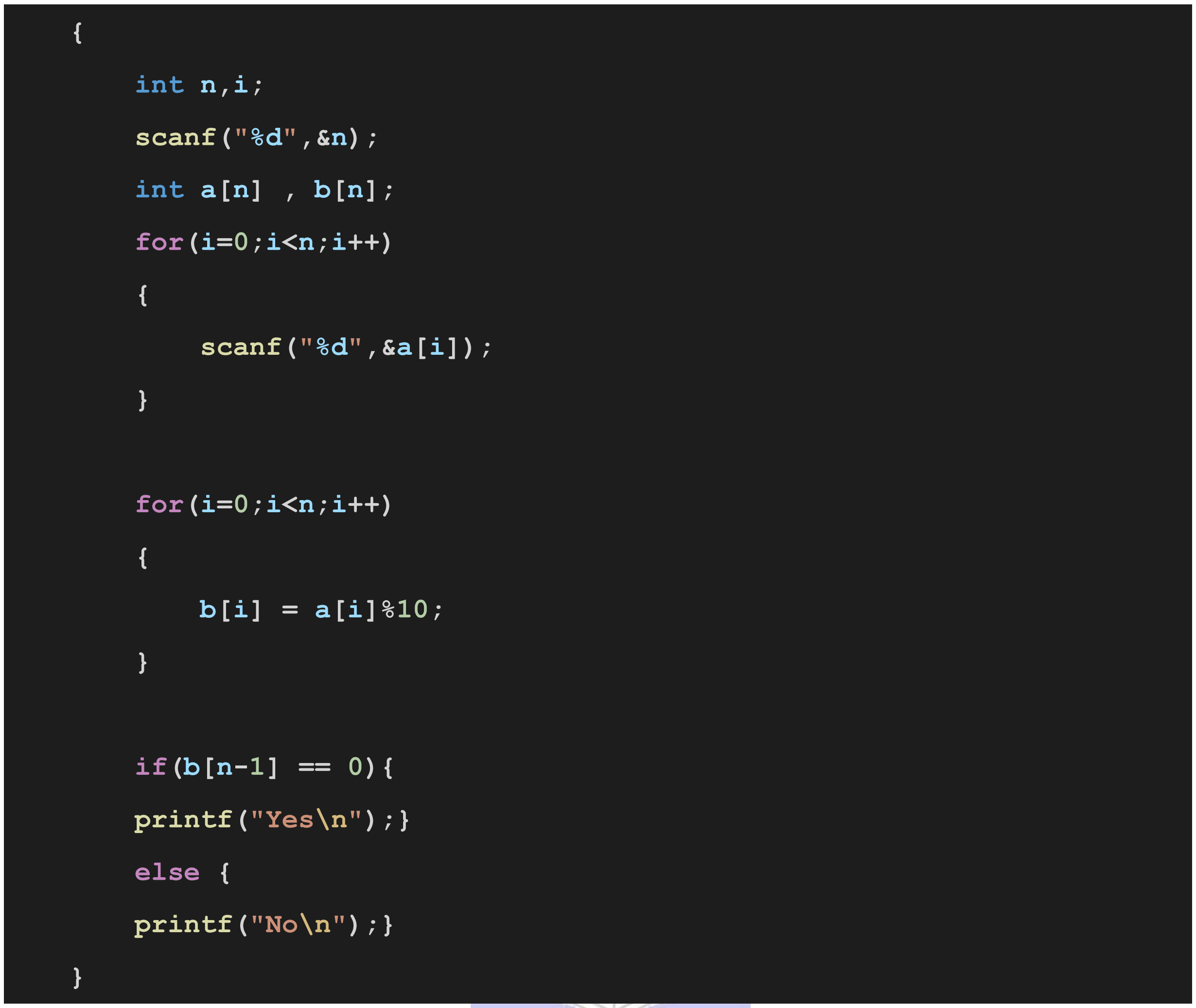
| **Sr. No.** | **Sample Input** | **Sample Output** | **Description** | **Test Case Type (general/special)** |
|---|---|---|---|---|
| 1. | 5<br>45 23 65 22 74 | No | array with 5 integer numbers | General |
| 2. | 6<br>13 60 24 12 10 130 | Yes | Array with 6 integer numbers | General |
| 3. | 4<br>13 60 -24 12 | No | Array with 3 positive integer and one negative integer | General |
| 4. | a | No output as after pressing enter nothing is displayed | Number of elements in character format | Special |
| 5 | a | No | Number of elements and n elements in character format | Special Case |
| 6 | -4 | Yes | Number of element is a negative integer and few n elements are also negative | General |

| | | | | |
|---|---|---|---|---|
| 7 | * | No | Number of element is a special character | Special |
| 8 | 6<br>* # $ ! ~ ^ | No | Array with special character | Special |
| 9 | <br>11 2 3 | No Output | Nothing is entered in the number of elements | Special |
| 10 | 4<br>123456 12 478878 2 | No | Array with greater than integer value | General |
| 11 | 3<br>1 | No output | Array with only one input element and two element as space | Special |
| 12 | 5<br>1,69,58,63,24 | No | Different elements contain comma in between them | General |
| 13 | 4, | No output | Number of element contains comma after the integer | General |
| 14 | 0 | No | Array is of zero size and two enters are placed simultaneously | Special |
| 15 | 4<br>0.1 2.2 3.89 6.666 | No | Array contains floating point numbers | General |
| 16 | 5.1 | No | Number of element is in floating point number | General |
| 17 | 5<br>12 32 22 20 | No output | Array elements is less than the number of elements | General |
| 18 | 4<br>12 65 98 20 60 | No<br>Yes | Array of elements is greater than the total number of elements | General |
| 19 | 5 23 52 52 1 20 | No<br>yes | Array of elements and number of elements are written on same line | Special |
| 20 | 5% | No | Number of element is written in percent format | Special |

**Solution:**

```
#include<stdio.h>

    int main()
```



```
{
    int n,i;
    scanf("%d",&n);
    int a[n] , b[n];
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }

    for(i=0;i<n;i++)
    {
        b[i] = a[i]%10;
    }

    if(b[n-1] == 0){
    printf("Yes\n");}
    else {
    printf("No\n");}
}
```

**Outcomes:**

**Post Lab Questions:**

Consider the given problem statement and related information:

**Problem**
You have been given a positive integer N where $1 \leq N \leq 12$. 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.

**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
    - Input value will be the integer.
    - The integer will be the number whose factorial needs to be carried out.

2. Constraints on input values
    - Here, Input value should be positive integer
    - It should be greater than or equal to 1 and less than or equal to 12.
    - It should not be a negative number
    - Not a floating number.
    - Number should be in range 0 to 32767.
    - It shouldn't be a character or a special character.
    - After entering the element user should press the Enter.

3. Output values
    - Output value will be the factorial of a number and it will be in the integer format.

4. Constraints on output values
    - Output value will return the single integer answer of the factorial of the number.

5. Specified format for input values
    - Input values should be a positive integer.
    - It should be in a specified range.
    - Users must press enter after entering the element.

6. Specified format for output values
    - It should only return a single value positive integer.

**Task 2:**
Identify general and special test cases for given problem statements. 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 | 5 | 120 | Entered the number as 5 | General |
| 2 | -9 | 1 | Entered the number in negative | General |
| 3 | c | 1 | Enter the Character as input | Special |
| 4 | 1.22 | 1 | Entered the input in float number | Special |
| 5 | 15 | 2004310016 | Entered the input greater than 12 | Special |
| 6 | 9! | 362880 | Entered the input in factorial format | Special |
| 7 | -9.6 | 1 | Entered the input in negative float number | Special |
| 8 | 1000 | 0 | Enter the input much greater than 12 | General |
| 9 | f52 | 1 | Entered the input as alphanumeric | Special |

**Code:**

```
#include <stdio.h>
void main(){
  int i,f=1,num;
  scanf("%d",&num);
  for(i=1;i<=num;i++)
  {
      f=f*i;
  }
  printf("%d",f);
}
```

---

**Conclusion: (Conclusion to be based on the objectives and outcomes achieved)**
Thus, I have identified the test cases for the given problem.

---

**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