PROBLEM SOLVING

(Solving Various Types of Problems)

Summer Internship Report Submitted in partial

fulfillment of the requirement for undergraduate degree

Of

Bachelor of Technology

In

Computer Science and Engineering

By

S.Anand

221710308051

(github.com/siddaanand**)**

Under the Guidance of

____

Assistant Professor

GITAM

(DEEMED TO BE UNIVERSITY)

VISAKHAPATNAM HYDERABAD BENGALURU

Department Of Computer Science and Engineering

GITAM School of Technology

GITAM (Deemed to be University)

Hyderabad-502329

July 2020

DECLARATION

I submitted this industrial training work entitled **“Solving Various Types of Problems**” to GITAM (Deemed To Be University), Hyderabad in partial fulfillment of the requirements for the award of the degree of “**Bachelor of Technology**” in “**Computer Science and Engineering**”. I declare that it was carried out independently by me under the guidance of **Mr.** , Asst. Professor, GITAM (Deemed To Be University), Hyderabad, India.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Place: Hyderabad S.Anand

Date: 12-07-2020 221710308051

GITAM(DEEMED TO BE UNIVERSITY)

Hyderabad-502329, India

Dated:

CERTIFICATE

This is to certify that the Industrial Training Report entitled **“Solving Various Types of Problems”** is being submitted by S.Anand (221710308051) in partial fulfillment of the requirement for the award of **Bachelor of Technology in Computer Science And Engineering** at GITAM (Deemed to Be University), Hyderabad during the academic year 2019-20.

It is faithful record work carried out by her at the **Computer Science And Engineering Department**, GITAM University Hyderabad Campus under my guidance and supervision.

Dr. S.Phani Kumar

Assistant Professor

Professor and HOD

ACKNOWLEDGEMENT

Apart from my effort, the success of this internship largely depends on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have helped me in the successful competition of this internship.

I would like to thank, respected **Dr. N. Siva Prasad**, Pro Vice-Chancellor, GITAM Hyderabad, and Dr. CH. Sanjay, Principal, GITAM Hyderabad.

I would like to thank respected **Prof. S. Phani Kumar**, Head of the Department of Computer Science Engineering for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present the internship report. It helped me a lot to realize what we study for.

I would like to thank the respected faculties who helped me to make this internship a successful accomplishment.

I would also like to thank my friends who helped me to make my work more organized and well-stacked till the end.

S. Anand

221710308051

TABLE OF CONTENTS

1. Introduction

Problem solving is the act of defining a problem; determining the cause of the problem; identifying, prioritizing, and selecting alternatives for a solution; and implementing a solution. Four problems which are listed below are of different complexity and require different approach and logics in order to achieve desired Output/ Solution

Philaland Coin Problem: In this problem we find out minimum number of denominations of coins required based on given input.
Counting Rock Samples: In this problem we find out the number of rocks present in each of the ranges as per given condition and based on given input.
Kth factor of n: In this problem we find out. largest factor of N.
Collecting Candies Problem: In this problem we the minimum time in which all the candies can be collected.

C is a general-purpose high-level language that was originally developed by Dennis Ritchie for the Unix operating system. It was first implemented on the Digital Equipment Corporation PDP- 11 computer in 1972. C was initially used for system development work, in particular the programs that make-up the operating system. C was adoped as a system development language because it produces code that runs nearly as fast as code written in assembly language.

2 . Problem 1

Philaland Coins Problem

2.1 Problem Statement

The problem solvers have found a new Island for coding and named it as Philaland.These smart people were given a task to make purchase of items at the Island easier by distributing various coins with different value.Manish has come up with a solution that if we make coins category starting from $1 till the maximum price of item present on Island, then we can purchase any item easily. He added following example to prove his point.

Let's suppose the maximum price of an item is 5$ then we can make coins of {$1, $2, $3, $4, $5}to purchase any item ranging from $1 till $5.

Now Manisha, being a keen observer suggested that we could actually minimize the number of coins required and gave following distribution {$1, $2, $3}. According to him any item can be purchased one time ranging from $1 to $5. Everyone was impressed with both of them.Your task is to help Manisha come up with minimum number of denominations for any arbitrary max price in Philaland.

- **Input Format**
  - First line contains an integer T denoting the number of test cases.
  - Next T lines contain an integer N denoting the maximum price of the item presents Philaland.

- **Output Format**
  - For each test case print a single line denoting the minimum number of denominations of coins required.

- **Constraints**
  - 1<=T<=100
  - 1<=N<=5000

Sample Input:

```
 2
10
 5
```

Sample Output:

```
 4
3
```

Explanation:

- For test case 1, N=10.
  - According to Manish {$1, $2, $3,… $10} must be distributed.
  - But as per Manisha only {$1, $2, $3, $4} coins are enough to purchase any item ranging from $1 to $10. Hence minimum is 4. Likewise denominations could also be {$1, $2, $3, $5}. Hence answer is still 4.

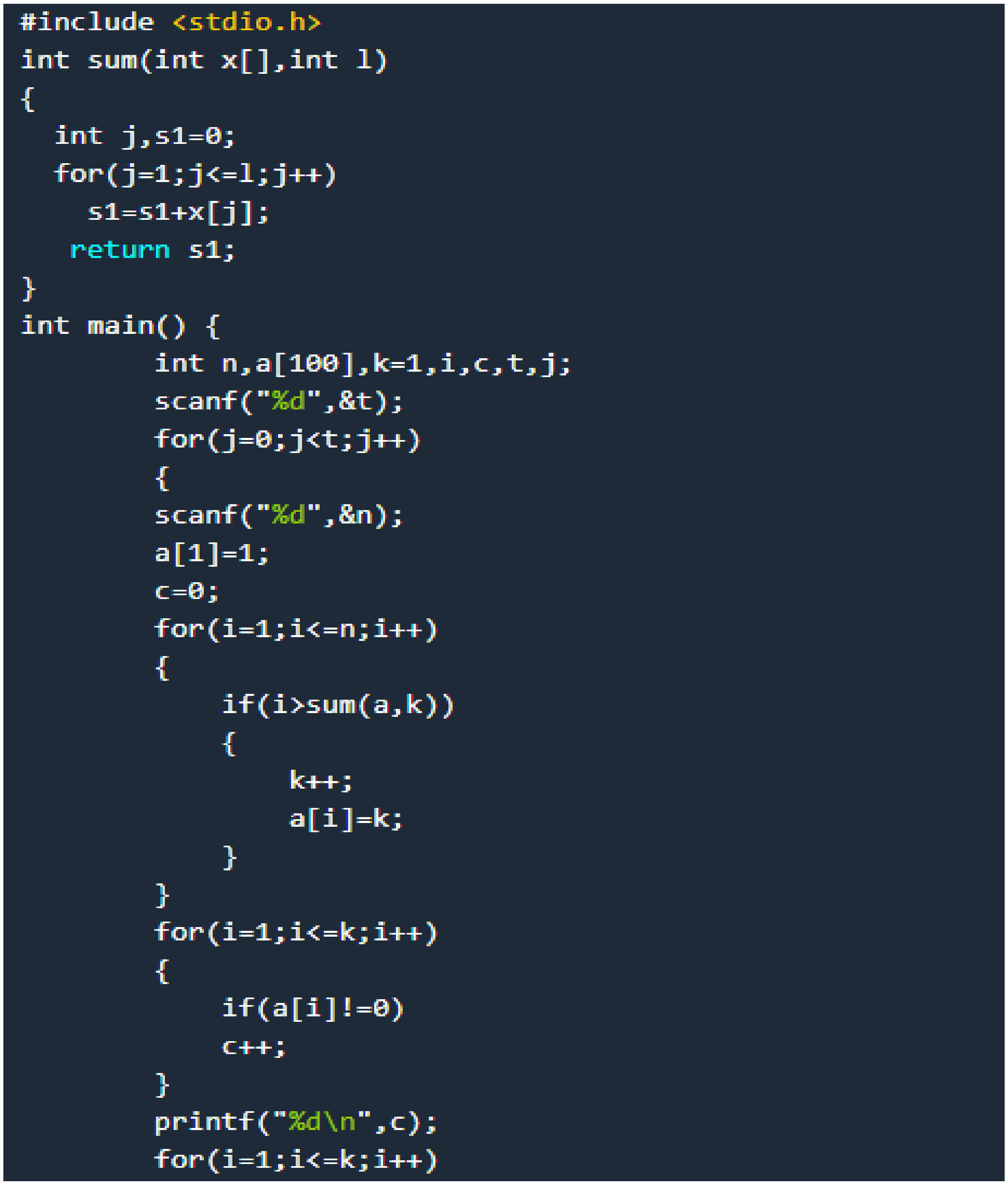
- For test case 2, N=5.
  - According to Manish {$1, $2, $3, $4, $5} must be distributed.
  - But as per Manisha only {$1, $2, $3} coins are enough to purchase any item ranging from $1 to $5. Hence minimum is 3. Likewise denominations could also be {$1, $2, $4}. Hence answer is still 3.

Concepts Used To Solve:

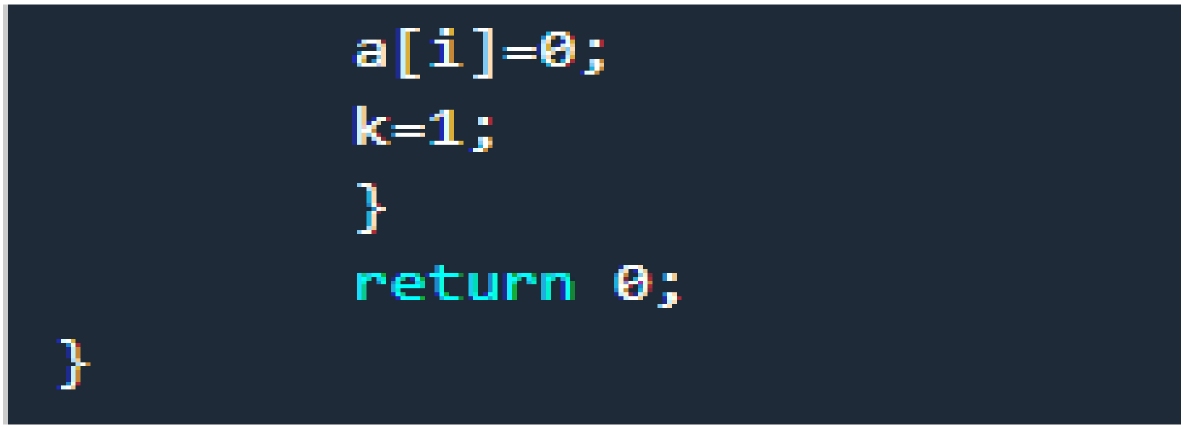
Arrays- An **array** is a collection of data items, all of the same type, accessed using a common name. A one-dimensional **array** is like a list; A two-dimensional **array** is like a table; The **C** language places no limits on the number of dimensions in an **array**, though specific implementations may.

Loops- Loops are control structures used to repeat a given section of code a certain number of times or until a particular condition is met.

2.2 Coding



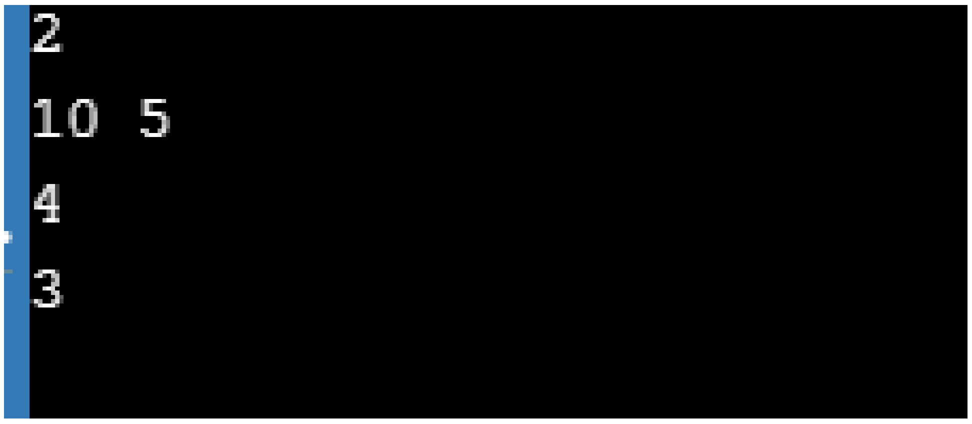
```
#include <stdio.h>
int sum(int x[],int l)
{
  int j,s1=0;
  for(j=1;j<=l;j++)
    s1=s1+x[j];
   return s1;
}
int main() {
        int n,a[100],k=1,i,c,t,j;
        scanf("%d",&t);
        for(j=0;j<t;j++)
        {
        scanf("%d",&n);
        a[1]=1;
        c=0;
        for(i=1;i<=n;i++)
        {
            if(i>sum(a,k))
            {
                k++;
                a[i]=k;
            }
        }
        for(i=1;i<=k;i++)
        {
            if(a[i]!=0)
            c++;
        }
        printf("%d\n",c);
        for(i=1;i<=k;i++)
```



```
            a[i]=0;
            k=1;
            }
            return 0;
}
```

Fig 2.2.1

2.3 Output



```
2
10 5
4
3
```

Fig 2.3.1

3 . Problem 2

Counting Rock Samples

3.1 Problem Statement

Juan Marquinho is a geologist and he needs to count rock samples in order to send it to a chemical laboratory. He has a problem: The laboratory only accepts rock samples by a range of its size in ppm (parts per million).

Juan Marquinho receives the rock samples one by one and he classifies the rock samples according to the range of the laboratory. This process is very hard because the number of rock samples may be in millions.

Juan Marquinho needs your help, your task is to develop a program to get the number of rocks in each of the ranges accept ed by the laboratory.

Input Format: An positive integer S (the number of rock samples) separated by a blank space, and a positive integer R (the number of ranges of the laboratory); A list of the sizes of S samples (in ppm), as positive integers separated by space R lines where the ith line containing two positive integers, space separated, indicating the minimum size and maximum size respectively of the ith range.

Output Format: R lines where the ith line contains a single non-negative integer indicating the number of the samples which lie in the ith range.

Constraints:

- 10 <= S <= 10000
- 1 <= R <= 1000000
- 1<=size of Sample <= 1000

Example 1

- Input: 10 2
- 345 604 321 433 704 470 808 718 517 811
- 300 350
- 400 700

Output: 2 4

Explanation:

There are 10 samples (S) and 2 ranges (R). The samples are 345, 604,811. The ranges are 300-350 and 400-700. There are 2 samples in the first range (345 and 321) and 4 samples in the second range (604, 433, 470, 517). Hence the two lines of the output are 2 and 4

Example 2

- Input: 20 3
- 921 107 270 631 926 543 589 520 595 93 873 424 759 537 458 614 725 842 575 195
- 1 100
- 50 600
- 1 1000

Output: 1 12 20

Explanation:

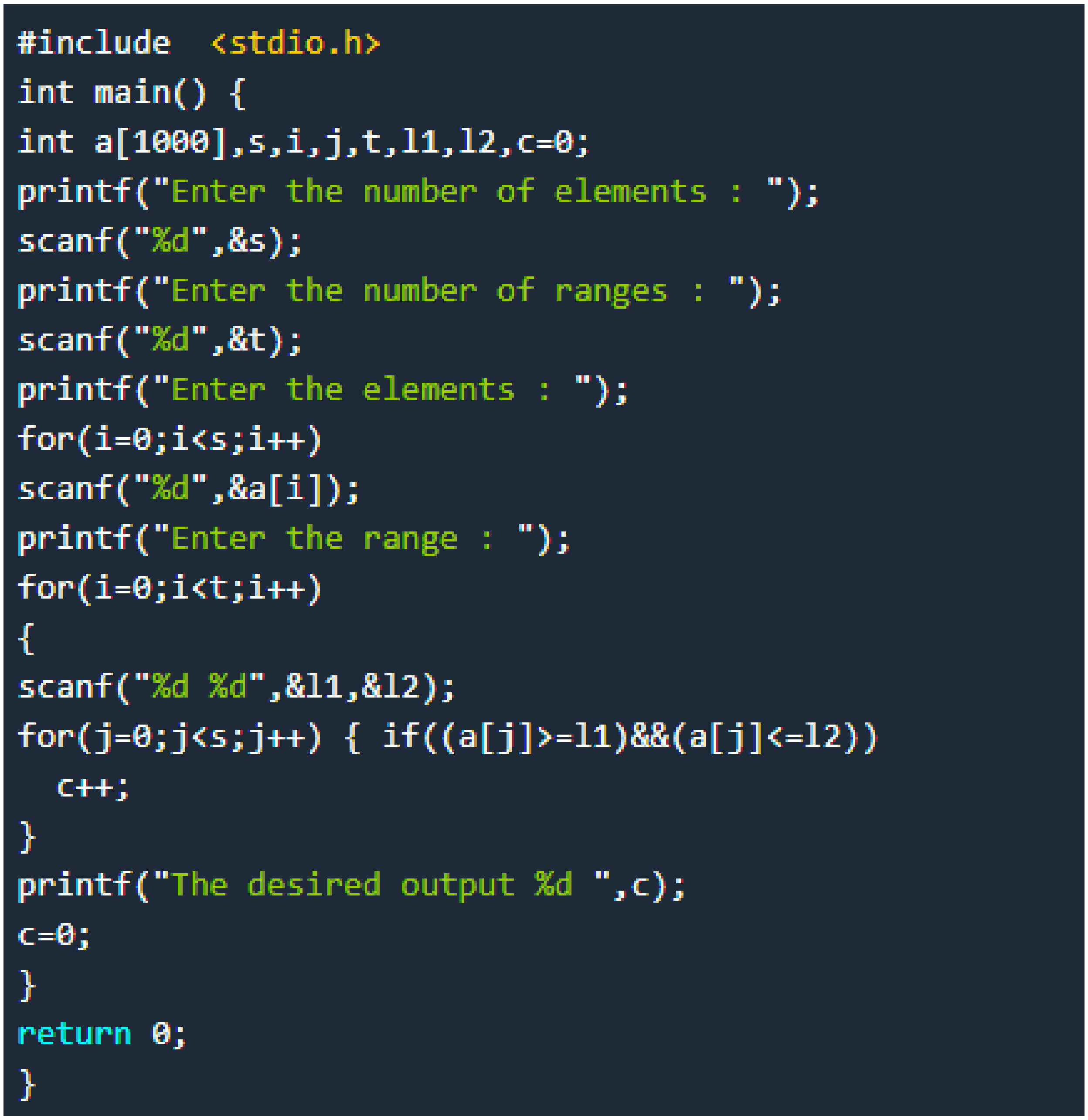
There are 20 samples and 3 ranges. The samples are 921, 107 195. The ranges are 1-100, 50-600 and 1-1000. Note that the ranges are overlapping. The number of samples in each of the three ranges is 1, 12 and 20 respectively. Hence the three lines of the output are 1, 12 and 20.

Concepts Used To Solve:

Arrays- An **array** is a collection of data items, all of the same type, accessed using a common name. A one-dimensional **array** is like a list; A two-dimensional **array** is like a table; The **C** language places no limits on the number of dimensions in an **array**, though specific implementations may.

Loops- Loops are control structures used to repeat a given section of code a certain number of times or until a particular condition is met.

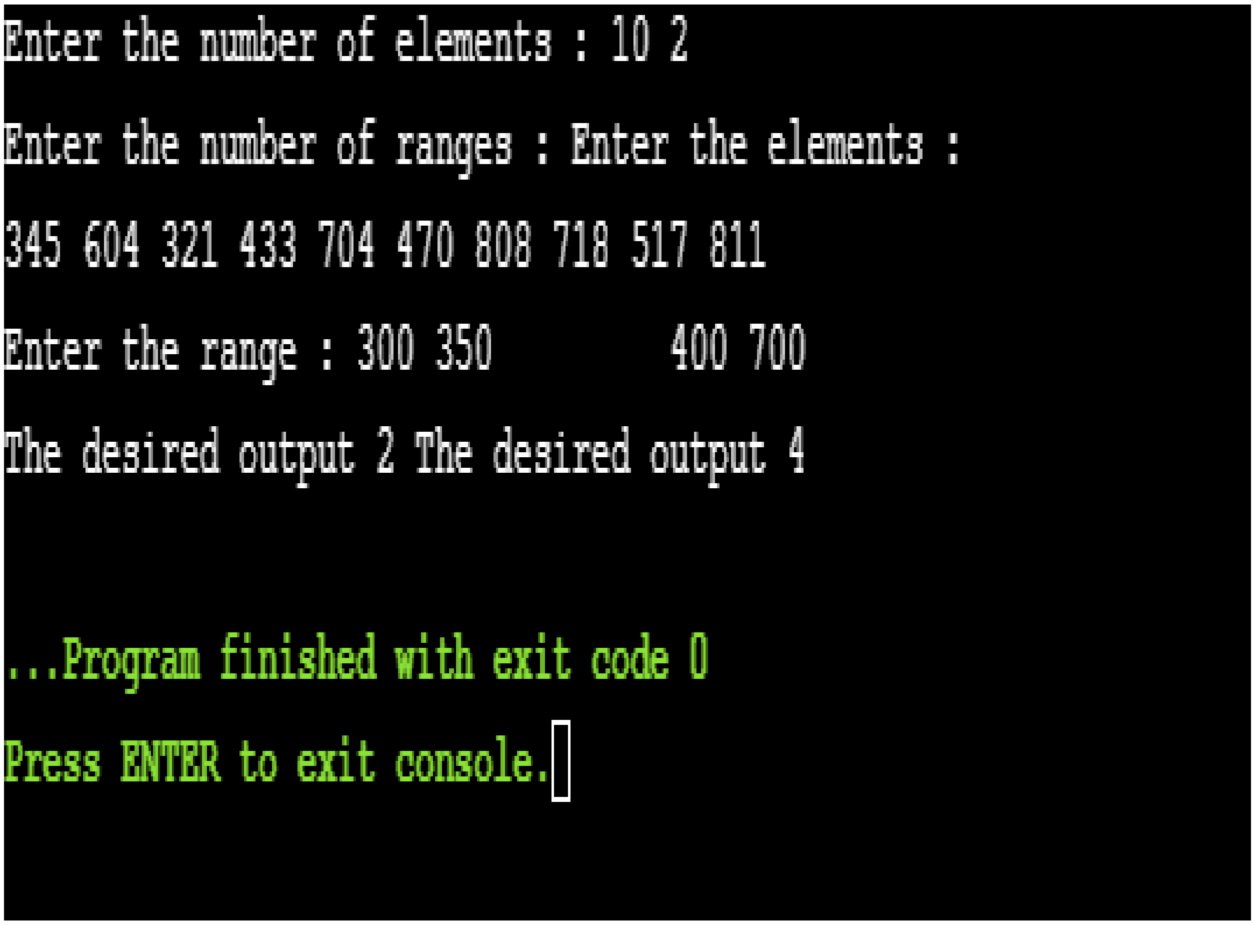
3.2 Coding



```c
#include  <stdio.h>
int main() {
int a[1000],s,i,j,t,l1,l2,c=0;
printf("Enter the number of elements : ");
scanf("%d",&s);
printf("Enter the number of ranges : ");
scanf("%d",&t);
printf("Enter the elements : ");
for(i=0;i<s;i++)
scanf("%d",&a[i]);
printf("Enter the range : ");
for(i=0;i<t;i++)
{
scanf("%d %d",&l1,&l2);
for(j=0;j<s;j++) { if((a[j]>=l1)&&(a[j]<=l2))
  c++;
}
printf("The desired output %d ",c);
c=0;
}
return 0;
}
```

Fig 3.2.1

3.3 Output



```
Enter the number of elements : 10 2

Enter the number of ranges : Enter the elements :

345 604 321 433 704 470 808 718 517 811

Enter the range : 300 350         400 700

The desired output 2 The desired output 4



...Program finished with exit code 0

Press ENTER to exit console.
```

Fig 3.3.1

4 . Problem 3

Kth factor of n

4.1 Problem Statement

A positive integer d is said to be a factor of another positive integer N if when N is divided by d, the remainder obtained is zero. For example, for number 12, there are 6 factors 1, 2, 3, 4, 6, 12. Every positive integer k has at least two factors, 1 and the number k itself. Given two positive integers N and k, write a program to print the kth largest factor of N.

Input Format: The input is a comma-separated list of positive integer pairs (N, k).

Output Format: The kth highest factor of N. If N does not have k factors, the output should be 1.

Constraints:

- $1<N<10000000000$
- $1<k<600$.

You can assume that N will have no prime factors which are larger than 13.

Example 1

- **Input**: 12,3
- **Output**: 4

Explanation: N is 12, k is 3. The factors of 12 are (1, 2, 3,4,6,12). The highest factor is 12 and the third largest factor is 4. The output must be 4.

Example 2

- **Input**: 30,9
- **Output**: 1

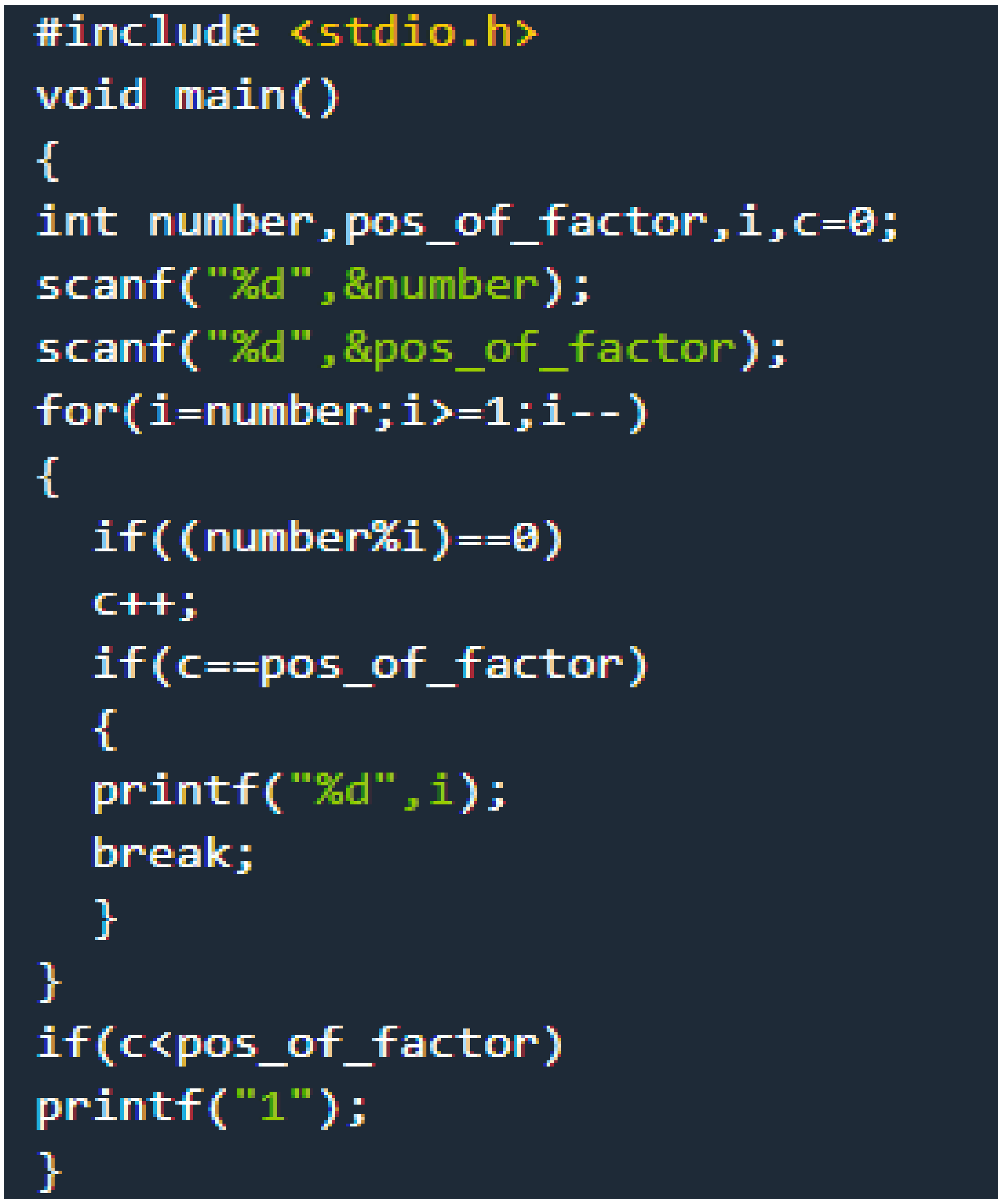
Explanation: N is 30, k is 9. The factors of 30 are (1, 2,3,5,6,10,15,30). There are only 8 factors. As k is more than the number of factors, the output

Concepts Used To Solve:

Arrays- An **array** is a collection of data items, all of the same type, accessed using a common name. A one-dimensional **array** is like a list; A two-dimensional **array** is like a table; The **C** language places no limits on the number of dimensions in an **array**, though specific implementations may.

Loops- Loops are control structures used to repeat a given section of code a certain number of times or until particular condition is met.

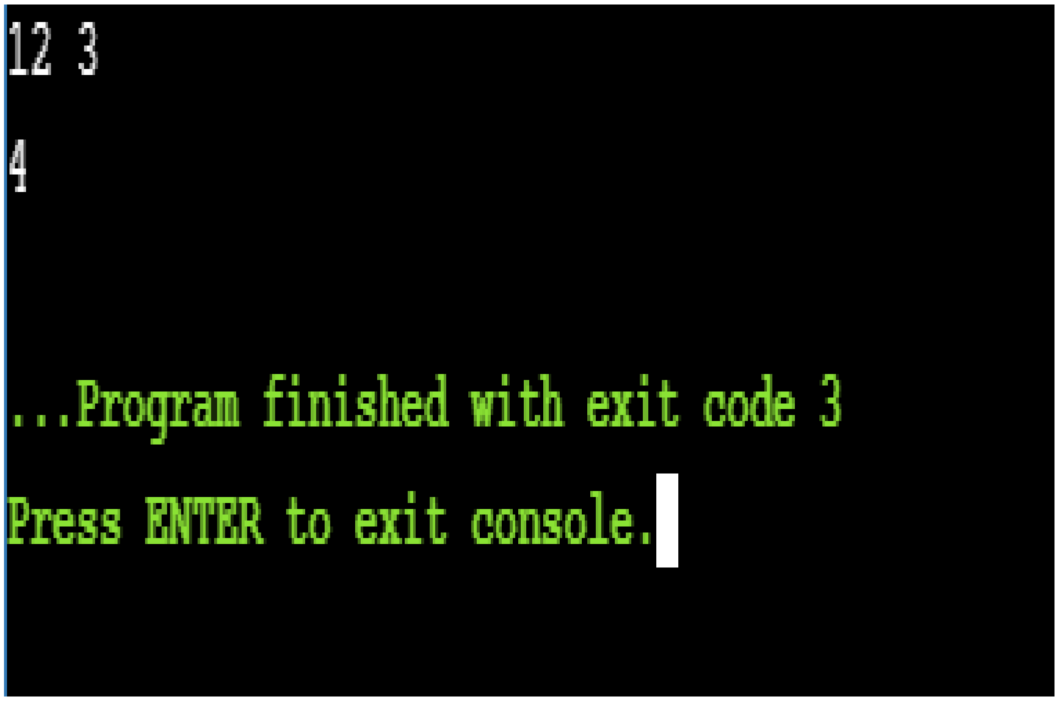
4.2 Coding



```
#include <stdio.h>
void main()
{
int number,pos_of_factor,i,c=0;
scanf("%d",&number);
scanf("%d",&pos_of_factor);
for(i=number;i>=1;i--)
{
   if((number%i)==0)
   c++;
   if(c==pos_of_factor)
   {
   printf("%d",i);
   break;
   }
}
if(c<pos_of_factor)
printf("1");
}
```

Fig 4.2.1

4.3 Output



```
12 3
4


...Program finished with exit code 3
Press ENTER to exit console.
```

Fig 4.3.1

1. **Problem 4**

Collecting Candies Problem

5.1 Problem Statement

Krishna loves candies a lot, so whenever he gets them, he stores them so that he can eat them later whenever he wants to.

He has recently received N boxes of candies each containing Ci candies where Ci represents the total number of candies in the ith box. Krishna wants to store them in a single box. The only constraint is that he can choose any two boxes and store their joint contents in an empty box only. Assume that there are an infinite number of empty boxes available.

At a time he can pick up any two boxes for transferring and if both the boxes contain X and Y number of candies respectively, then it takes him exactly X+Y seconds of time. As he is too eager to collect all of them he has approached you to tell him the minimum time in which all the candies can be collected.

Input Format:

- The first line of input is the number of test case T
- Each test case is comprised of two inputs
- The first input of a test case is the number of boxes N
- The second input is N integers delimited by whitespace denoting the number of candies in each box

Output Format: Print minimum time required, in seconds, for each of the test cases. Print each output on a new line.

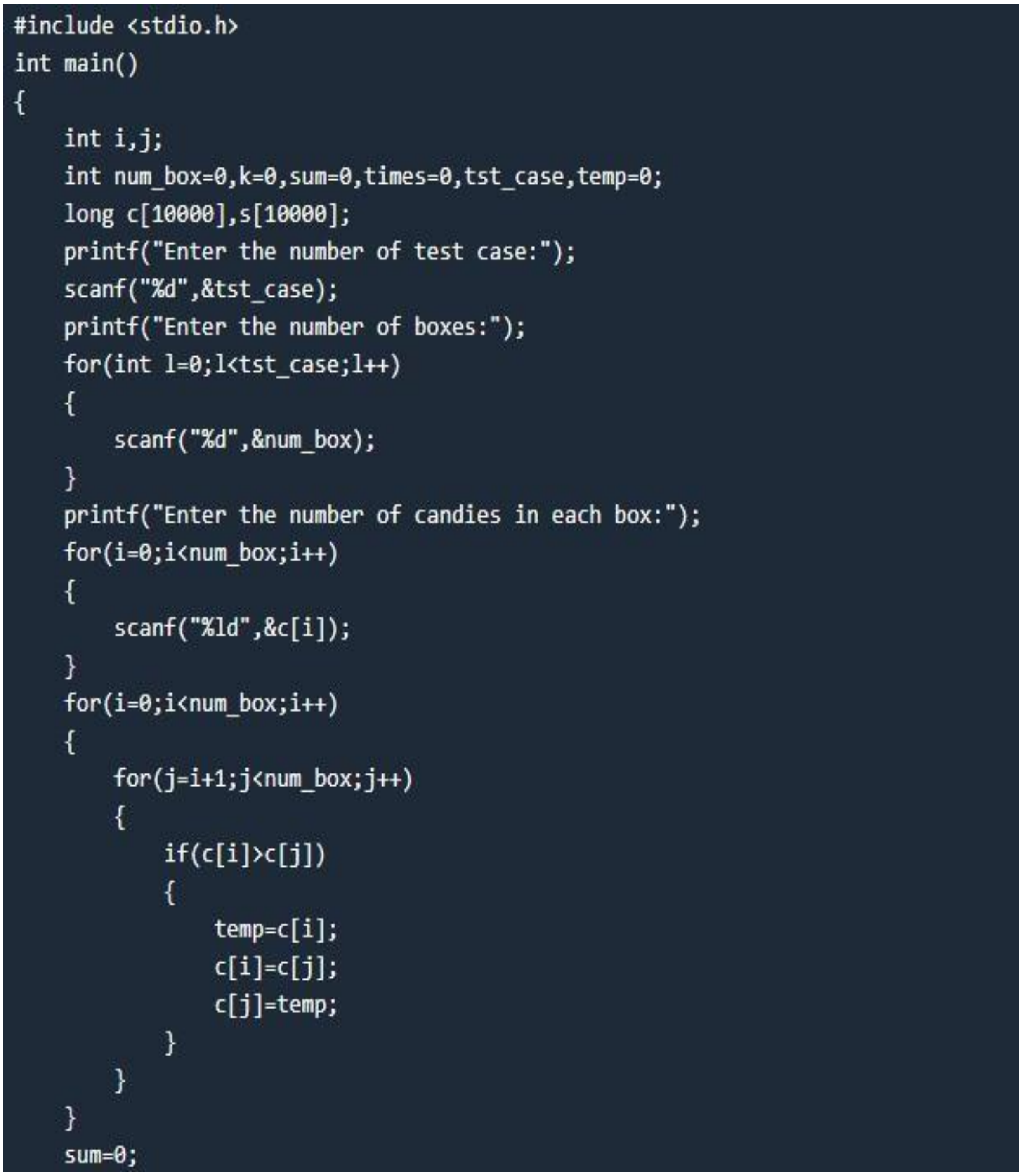
Constraints:

- 1 < T < 10
- 1 < N< 10000
- 1 < [Candies in each box] < 100009

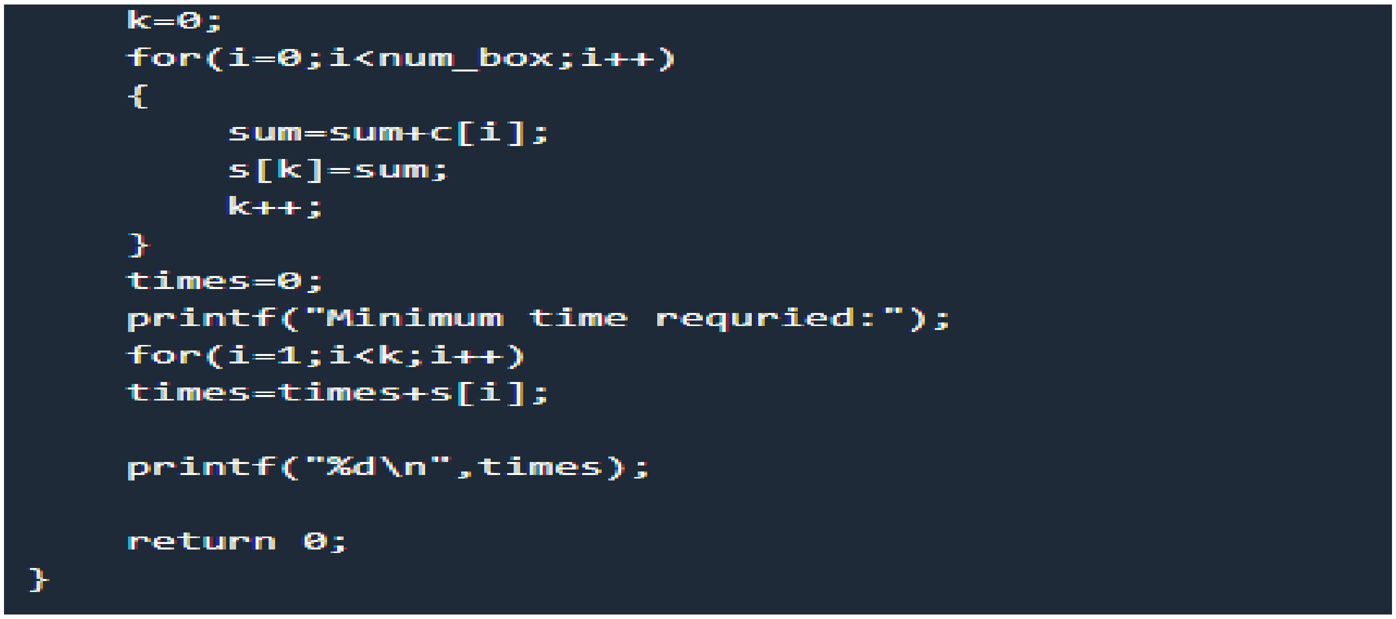
Concepts Used To Solve:

Arrays, For loop, Validation of Conditions

5.2 Coding



```c
#include <stdio.h>
int main()
{
    int i,j;
    int num_box=0,k=0,sum=0,times=0,tst_case,temp=0;
    long c[10000],s[10000];
    printf("Enter the number of test case:");
    scanf("%d",&tst_case);
    printf("Enter the number of boxes:");
    for(int l=0;l<tst_case;l++)
    {
        scanf("%d",&num_box);
    }
    printf("Enter the number of candies in each box:");
    for(i=0;i<num_box;i++)
    {
        scanf("%ld",&c[i]);
    }
    for(i=0;i<num_box;i++)
    {
        for(j=i+1;j<num_box;j++)
        {
            if(c[i]>c[j])
            {
                temp=c[i];
                c[i]=c[j];
                c[j]=temp;
            }
        }
    }
    sum=0;
```



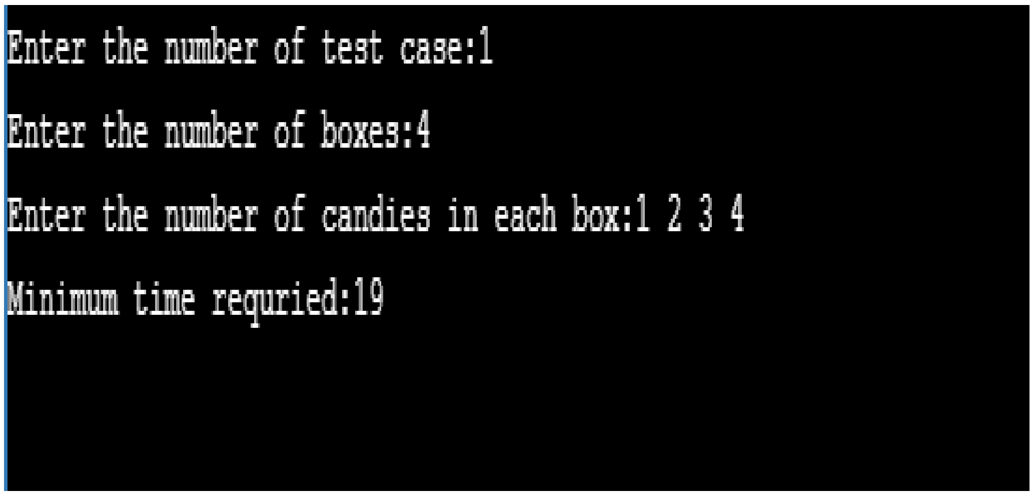
```
    k=0;
    for(i=0;i<num_box;i++)
    {
        sum=sum+c[i];
        s[k]=sum;
        k++;
    }
    times=0;
    printf("Minimum time requried:");
    for(i=1;i<k;i++)
    times=times+s[i];

    printf("%d\n",times);

    return 0;
}
```

Fig 5.2.1

5.3 Output


```
Enter the number of test case:1

Enter the number of boxes:4

Enter the number of candies in each box:1 2 3 4

Minimum time requried:19
```

Fig 5.3.1

2. Software Requirements

6.1 Hardware Requirements

This project can be executed in any system or an android phone without prior to any platform. We can use any online compiler and interpreter.

6.2 Software Requirements

There are two ways to execute this projects

1. Online compilers
2. Softwares for execution (DEV C++, ANACONDA…..)

Online Compilers require only internet connection. We have many free compilers with which we can code.

Softwares for execution need to be installed based on the user’s system specification. These help us to completely execute the project. These softwares are based on the platforms

7. REFERENCES

https://prepinsta.com/tcs-codevita/

https://www.faceprep.in/tcs/tcs-codevita-questions/

https://www.programminggeek.in/