



COMPANY SPECIFIC SERIES
ACCENTURE – CODING - TRAINER HANDOUT

PROGRAMS ASKED IN RECENT ACCENTURE DRIVES:

1. PROGRAM 1 - EDWARD'S BIRTHDAY

It is Edward's birthday today. His friends have bought him a huge circular cake. Edward wants to find out the maximum number of pieces he can get by making exactly N straight vertical cuts on the cake.

Your task is to write a function that returns the maximum number of pieces that can be obtained by making N number of cuts.

Note: Since the answer can be quite large, modulo it by 1000000007

Input Specification:

input1: An integer N denoting the number of cuts

Output Specification:

Return the maximum number of pieces that can be obtained by making N cuts on the cake

Example:

Input	Output	Explanation
1	2	Given the above scenario, if we make 1 cut on the cake, the maximum number of pieces of cake we obtain is 2 and $2 \% 1000000007 = 2$. Hence, 2 is returned as output.
5	16	Given the above scenario, if we make 5 cuts on the cake, the maximum number of pieces of cake we obtain is 16 and $16 \% 1000000007 = 16$. Hence, 16 is returned as output.





Code Solution in C

```
#include<stdio.h>
int maximumCuts(int n){
const unsigned int M = 1000000007;
int sum=1;
while(n!=0)
sum+=n--;
return sum%M;
}
int main(){
int n;
scanf("%d",&n);
int ans = maximumCuts(n);
printf("%d\n",ans);
return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int maximumCuts(int n){
const unsigned int M = 1000000007;
int sum=1;
while(n!=0)
sum+=n--;
return sum%M;
}
int main(){
int n;
cin >> n;
int ans = maximumCuts(n);
cout << ans << "\n";
return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
import java.lang.*;
import java.io.*;
class Main
{
```





```
public static void main (String[] args) throws IOException
{
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
Solution ob = new Solution();
System.out.println(ob.maximumCuts(n));
}
}

class Solution
{
static long maximumCuts(int n)
{
long M = 1000000007;
if(n==1 )
{
return 2;
}
if(n==0)
{
return 1;
}
return (n + maximumCuts(n-1))%M;
}
}
```

Code Solution in Python

```
class Solution:
def maximumCuts(self, n):
M = 1000000007
num=((n*(n+1))/2) +1
return num % M
if __name__ == '__main__':
n = int(input())
ob = Solution()
ans = ob.maximumCuts(n)
print(ans)
```





2. PROGRAM 2 - REGIONS ON A PLANE

Problem statement

Mr. Professor is a great scientist, but he is not able to find a solution to one problem. There are N straight lines that are not parallel, and no three lines go through the same point. The lines divide the plane into M regions. Write a function to find out the maximum number of such regions he can get on the plane.

Input Specification:

Input1: An integer N representing the number of straight lines ($0 \leq N \leq 100$)

Output Specification:

Return the maximum number of regions

Example:

Input	Output	Explanation
2	4	Given the above scenario, 2 lines divide the plane into 4 regions. Therefore, 4 is returned as the output.
3	7	Given the above scenario, 3 lines divide the plane into 7 regions. Therefore, 7 is returned as the output





Code Solution in C

```
#include<stdio.h>
void maxRegions(int n)
{
int num;
num = n * (n + 1) / 2 + 1;
printf("%d",num);
}
int main()
{
int n;
scanf("%d",&n);
maxRegions(n);
return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
void maxRegions(int n)
{
int num;
num = n * (n + 1) / 2 + 1;
cout << num;
}
int main(){
int n;
cin >> n;
maxRegions(n);
return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
import java.lang.*;
import java.io.*;
class Main
{
public static void main (String[] args) throws IOException
{
```





```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
Solution ob = new Solution();
ob.maxRegions(n);
}
}
class Solution
{
public static void maxRegions(int n)
{
int num;
num = n * (n + 1) / 2 + 1;
System.out.println(num);
}
}
```

Code Solution in Python

```
class Solution:
def maxRegions(self, n):
num = n * (n + 1) // 2 + 1
return num
if __name__ == '__main__':
n = int(input())
ob = Solution()
ans = ob.maxRegions(n)
print(ans)
```





3. PROGRAM 3 - SUM OF ODD INTEGERS IN ARRAY

Problem statement

An odd number is an integer which is not a multiple of 2.

You are required to implement the following function:

Int SumOddIntegers(int arr[], int n);

The function accepts an integer array ‘arr’ of length ‘n’ as its argument. You are required to calculate the sum of all odd integers in an array ‘an’ and return the same

Note:

Array can have negative integers

$n > 0$

Computed values lie within integer range

Example:

Input:

arr: 1 4 6 7 10 12 11 5

n: 8

Output:

24

Explanation:

The odd integers in array {1, 4, 6, 7, 10, 12, 11, 5} are {1, 7, 11, 5} and their sum is 24, hence 24 is returned.

The custom input format for the above case:

8

1 4 6 7 10 12 11 5

(The first line represents ‘n’ the second line represents the elements of the array)

Sample Input

arr: 2 4 9 7 11 13 25 31 6 8 10 24

n: 12

Sample Output

96

The custom input format for the above case:

12

2 4 9 7 11 13 25 31 6 8 10 24

(The first line represents ‘n’ the second line represents the elements of the array)





Code Solution in C

```
1 #include <stdio.h>
2 int sumAllOdds(int arr[], int size) {
3     int sum = 0;
4     for(int i = 0 ; i < size; i++) {
5         if(arr[i] % 2 != 0) { // checks if number is odd
6             sum += arr[i];
7         }
8     }
9     return sum;
10 }
11
12 int main() {
13     int n;
14     scanf("%d",&n);
15     int a[n];
16     for(int i=0;i<n;i++)
17     {
18         scanf("%d",&a[i]);
19     }
20     printf("%d\n",sumAllOdds(a,n));
21     return 0;
22 }
```

Code Solution in C++

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int sumAllOdds(int arr[], int size) {
4     int sum = 0;
5     for(int i = 0 ; i < size; i++) {
6         if(arr[i] % 2 != 0) { // checks if number is odd
7             sum += arr[i];
8         }
9     }
10    return sum;
11 }
12
13 int main() {
14     int n;
15     cin>>n;
16     int a[n];
17     for(int i=0;i<n;i++)
18     {
19         cin>>a[i];
20     }
21     cout<<sumAllOdds(a,n)<<endl;
22     return 0;
23 }
```





Code Solution in JAVA

```
1 import java.util.*;
2 import java.lang.*;
3 import java.io.*;
4
5 class sumAllOdds{
6     public static void main(String args[])
7     {
8         int odd = 0;
9         Scanner in = new Scanner(System.in);
10        int n = in.nextInt();
11        int[] arr = new int[n];
12        for (int i = 0; i <n; i++){
13            arr[i] = in.nextInt();
14        }
15        for (int i = 0; i <n; i++) {
16            if ((arr[i] % 2) != 0)
17                odd += arr[i];
18        }
19        System.out.println(odd);
20    }
21 }
```





4. PROGRAM 4 - INVERSION COUNT IN ARRAY

Problem statement

Let j and k be two indices in an array A.

If $j < K$ and $A[j] > A[k]$, then the pair (j,k) is known as an “Inversion pair”.

You are required to implement the following function:

```
int InversionCount(int *A, int n);
```

The function accepts an array ‘A’ of ‘n’ unique integers as its argument. You are required to Calculate the number of ‘Inversion pair’ in an array A, and return.

Note:

If ‘A’ is NULL (None in case of python). return – 1

I_f_n_ ≤ 2,_r_e_t_u_r_n_0_

Example:

Input:

A: 1 20 6 4 5

n: 5

Output:

5

Explanation:

The inversion pair in array A are (20,6),(20,4),(20,5),(6,4) and (6,5), the count of the inversions are 5, hence 5 is returned.

The custom input format for the above case:

5

1 20 6 4 5

(The first line represents the size of the array, the second line represents the elements of the array)

Sample Input

A: 13 10 9 6 21 15 14

n: 7

Sample Output

9

The custom input format for the above case:

7

13 10 9 6 21 15 14





Code Solution in C

```
1 #include <stdio.h>
2
3 int getInvCount(int arr[], int n)
4 {
5     int inv_count = 0;
6     for (int i = 0; i < n - 1; i++)
7         for (int j = i + 1; j < n; j++)
8             if (arr[i] > arr[j])
9                 inv_count++;
10
11    return inv_count;
12 }
13
14 int main() {
15     int n;
16     scanf("%d", &n);
17     int a[n];
18     for (int i=0;i<n;i++)
19     {
20         scanf("%d", &a[i]);
21     }
22     printf("%d\n", getInvCount(a, n));
23     return 0;
24 }
```

Code Solution in C++

```
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 int getInvCount(int arr[], int n)
5 {
6     int inv_count = 0;
7     for (int i = 0; i < n - 1; i++)
8         for (int j = i + 1; j < n; j++)
9             if (arr[i] > arr[j])
10                 inv_count++;
11
12    return inv_count;
13 }
14
15 int main() {
16     int n;
17     cin>>n;
18     int a[n];
19     for (int i=0;i<n;i++)
20     {
21         cin>>a[i];
22     }
23     cout<< getInvCount(a, n)<<endl;
24     return 0;
25 }
```





Code Solution in JAVA

```
1 import java.util.*;
2 import java.lang.*;
3 import java.io.*;
4
5 class InvCount {
6     public static void main(String[] args)
7     {
8         Scanner in = new Scanner(System.in);
9         int n = in.nextInt();
10        int[] arr = new int[n];
11        for (int i = 0; i <n; i++){
12            arr[i] = in.nextInt();
13        }
14        int inv_count = 0;
15        for (int i = 0; i < n - 1; i++)
16            for (int j = i + 1; j < n; j++)
17                if (arr[i] > arr[j])
18                    inv_count++;
19        System.out.println(inv_count);
20    }
21 }
```





5. PROGRAM 5 - SUPERIOR ARRAY ELEMENT

Problem statement

In an array a superior element is one which is greater than all elements to its right.

The rightmost element will always be considered as a superior element.

You are given a function,

```
Int FindNumberOfSuperiorElements(int* arr, int n);
```

The function accepts an integer array ‘arr’ and its length ‘n’. Implement the function to find and return the number of superior elements in array ‘arr’.

Assumptions:

1. N > 0.
2. Array index starts from 0.

Example:

Input	Output	Explanation
Arr: 7 9 5 2 8 7	3	9 is greater than all the elements to its right, 8 is greater than element to its right and 7 is the rightmost element. Hence total 3 superior elements.

The Custom input format for the above case:

6

7 9 5 2 8 7

(The first line represents ‘n’ the second line represents the elements of the array ‘arr’)

Sample Input	Output
Arr: 2 8 9 7 4 2	4

The custom input format for the above case:

Input:

6

2 8 9 7 4 2

(The first line represents ‘n’, the second line represents the elements of the array ‘arr’)

Output:

4

Explanation: 9 is greater than all the elements to its right, 7 is greater than element to its right, 4 is greater than element to its right and 2 is the rightmost element. Hence total 4 superior elements.





Code Solution in C

```
1 #include<stdio.h>
2 int CountSuperior (int arr[], int size)
3 {
4     int ans=0,i;
5     for ( i = 0; i < size; i++)
6     {
7         int j;
8         for (j = i+1; j < size; j++)
9         {
10            if (arr[i] <=arr[j])
11                break;
12        }
13        if (j == size) // the loop didn't break
14            ans++;
15    }
16 return ans;
17 }
18
19 int main()
20 {
21
22     int n,i;
23     scanf("%d",&n);
24     int arr[n];
25     for ( i = 0; i < n; i++)
26         scanf("%d",&arr[i]);
27     printf("%d",CountSuperior(arr,n));
28     return 0;
29 }
```

Code Solution in C++

```
1 #include<iostream>
2 using namespace std;
3 int CountSuperior (int arr[], int size)
4 {
5     int ans=0;
6     for (int i = 0; i < size; i++)
7     {
8         int j;
9         for (j = i+1; j < size; j++)
10        {
11            if (arr[i] <=arr[j])
12                break;
13        }
14        if (j == size) // the loop didn't break
15            ans++;
16    }
17 return ans;
18 }
19
20 int main()
21 {
22
23     int n;;
24     cin>>n;
25     int arr[n];
26     for (int i = 0; i < n; i++)
27         cin>>arr[i];
28     cout<<CountSuperior(arr,n);
29     return 0;
30 }
```





Code Solution in JAVA

```
1 import java.io.*;
2 import java.util.*;
3 class CountSuperior
4 {
5
6     int NoOfSuperior(int arr[], int size)
7     {
8         int ans=0;
9         for (int i = 0; i < size; i++)
10        {
11            int j;
12            for (j = i + 1; j < size; j++)
13            {
14                if (arr[i] <=arr[j])
15                    break;
16            }
17            if (j == size) // the loop didn't break
18                ans++;
19        }
20        return ans;
21    }
22
23
24    public static void main(String[] args)
25    {
26        CountSuperior superior= new CountSuperior();
27        int n;
28        Scanner sc=new Scanner(System.in);
29        n=sc.nextInt();
30        int[] arr = new int[n];
31        for(int i=0; i<n; i++)
32        {
33            arr[i]=sc.nextInt();
34        }
35
36        System.out.print(superior.NoOfSuperior(arr, n));
37    }
38 }
39 }
```

Code Solution in PYTHON

```
1 arr=list(map(int, input().split()))
2 c=0
3 for i in range(len(arr)-1):
4     j=i+1
5     arr_n=arr[j:]
6     m=max(arr_n)
7     if arr[i] > m:
8         c+=1
9 print(c+1)
```





6. PROGRAM 6 - RAT COUNT HOUSE

Problem Description:

The function accepts two positive integers ‘r’ and ‘unit’ and a positive integer array ‘arr’ of size ‘n’ as its argument ‘r’ represents the number of rats present in an area, ‘unit’ is the amount of food each rat consumes and each ith element of array ‘arr’ represents the amount of food present in ‘i+1’ house number, where $0 \leq i < n$

Note:

- Return -1 if the array is null
- Return 0 if the total amount of food from all houses is not sufficient for all the rats.
- Computed values lie within the integer range.

Example:

Input:

- r: 7
- unit: 2
- n: 8
- arr: 2 8 3 5 7 4 1 2

Output: 4

Explanation:

Total amount of food required for all rats = $r * \text{unit}$

$$= 7 * 2 = 14$$

The amount of food in 1st houses = $2+8+3+5 = 18$. Since, the amount of food in 1st 4 houses is sufficient for all the rats. Thus, output is 4.





Code Solution in C

```
#include <stdio.h>

int calculate (int r, int unit, int arr[], int n)
{
    if (n == 0)
        return -1;
    int totalFoodRequired = r * unit;
    int foodTillNow = 0;
    int house = 0;
    for (house = 0; house < n; ++house)
    {
        foodTillNow += arr[house];
        if (foodTillNow >= totalFoodRequired)
        {
            break;
        }
    }
    if (totalFoodRequired > foodTillNow)
        return 0;
    return house + 1;
}

int main ()
{
    int r;
    scanf ("%d", &r);
    int unit;
    scanf ("%d", &unit);
    int n;
    scanf ("%d", &n);
    int arr[n];
    for (int i = 0; i < n; ++i)
    {
        scanf ("%d", &arr[i]);
    }
    printf ("%d", calculate (r, unit, arr, n));
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int calculate (int r, int unit, int arr[], int n)
{
    if (n == 0)
        return -1;
    int totalFoodRequired = r * unit;
    int foodTillNow = 0;
    int house = 0;
    for (house = 0; house < n; ++house)
    {
        foodTillNow += arr[house];
        if (foodTillNow >= totalFoodRequired)
        {
            break;
        }
    }
    if (totalFoodRequired > foodTillNow)
        return 0;
    return house + 1;
}
int main ()
{
    int r;
    cin >> r;
    int unit;
    cin >> unit;
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n; ++i)
    {
        cin >> arr[i];
    }
    cout << calculate (r, unit, arr, n);
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int solve (int r, int unit, int arr[], int n)
    {
        if (arr == null)
            return -1;
        int res = r * unit;
        int sum = 0;
        int count = 0;
        for (int i = 0; i < n; i++)
        {
            sum = sum + arr[i];
            count++;
            if (sum >= res)
                break;
        }
        if (sum < res)
            return 0;
        return count;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int r = sc.nextInt ();
        int unit = sc.nextInt ();
        int n = sc.nextInt ();
        int arr[] = new int[n];

        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt ();
        System.out.println (solve (r, unit, arr, n));
    }
}
```

Code Solution in PYTHON

```
def calculate (r, unit, arr, n):
    if n == 0:
        return -1

    totalFoodRequired = r * unit
    foodTillNow = 0
    house = 0

    for house in range (n):
        foodTillNow += arr[house]
        if foodTillNow>=totalFoodRequired:
            break
    if totalFoodRequired > foodTillNow:
        return 0
    return house + 1

r = int (input ())
unit = int (input ())
n = int (input ())

arr = list (map (int, input ().split ()))
print (calculate (r, unit, arr, n))
```





7. PROGRAM 7 - BINARY STRING

Problem Description:

The Binary number system only uses two digits, 0 and 1 and the number system can be called binary string. You are required to implement the following function:

```
int OperationsBinaryString(char* str);
```

The function accepts a string str as its argument. The string str consists of binary digits separated with an alphabet as follows:

- – A denotes AND operation
- – B denotes OR operation
- – C denotes XOR Operation

You are required to calculate the result of the string str, scanning the string to right taking one operation at a time, and return the same.

Note:

- No order of priorities of operations is required
- Length of str is odd
- If str is NULL or None (in case of Python), return -1

Input:

str: 1C0C1C1A0B1

Output:

1

Explanation:

The alphabets in str when expanded becomes “1 XOR 0 XOR 1 XOR 1 AND 0 OR 1”, result of the expression becomes 1, hence 1 is returned.

Sample Input:

0C1A1B1C1C1B0A0

Output:

0





Code Solution in C

```
#include <stdio.h>
#include <string.h>

int OperationsBinaryString (char *str)
{
    if (str == NULL)
        return -1;
    int i = 1;
    int a = *str - '0';
    str++;
    while (*str != '\0')
    {
        char p = *str;
        str++;
        if (p == 'A')
            a &= (*str - '0');
        else if (p == 'B')
            a |= (*str - '0');
        else
            a ^= (*str - '0');
        str++;
    }
    return a;
}

int main ()
{
    char str[100];
    fgets (str, sizeof (str), stdin);
    int len = strlen (str);
    if (str[len - 1] == '\n')
    {
        str[len - 1] = '\0';           // Remove the newline character
        len--;                      // Decrement the length
    }
    int result = OperationsBinaryString (str);
    printf ("%d\n", result);
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int OperationsBinaryString (char *str)
{
    if (str == NULL)
        return -1;
    int i = 1;
    int a = *str - '0';
    str++;
    while (*str != '\0')
    {
        char p = *str;
        str++;
        if (p == 'A')
            a &= (*str - '0');
        else if (p == 'B')
            a |= (*str - '0');
        else
            a ^= (*str - '0');
        str++;
    }
    return a;
}
int main ()
{
    string s;
    getline (cin, s);
    int len = s.size ();
    char *str = &s[0];
    cout << OperationsBinaryString (str);
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int operationsBinaryString (String str)
    {
        if (str == null)
            return -1;
        int res = str.charAt (0) - '0';
        for (int i = 1; i < str.length ();)
        {
            char prev = str.charAt (i);
            i++;
            if (prev == 'A')
                res = res & (str.charAt (i) - '0');
            else if (prev == 'B')
                res = res | (str.charAt (i) - '0');
            else
                res = res ^ (str.charAt (i) - '0');
            i++;
        }
        return res;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        String str = sc.next ();
        System.out.println (operationsBinaryString (str));
    }
}
```

Code Solution in PYTHON

```
def OperationsBinaryString(str):
    a=int(str[0])
    i=1
    while i<len(str):
        if str[i]=='A':
            a&=int(str[i+1])
        elif str[i]=='B':
            a|=int(str[i+1])
        else:
            a^=int(str[i+1])
        i+=2
    return a
str=input()
print(OperationsBinaryString(str))
```





8. PROGRAM 8 - PASSWORD CHECKER

You are given a function.

```
int CheckPassword(char str[], int n);
```

The function accepts string str of size n as an argument. Implement the function which returns 1 if given string str is valid password else 0.

str is a valid password if it satisfies the below conditions.

- At least 4 characters
- At least one numeric digit
- At Least one Capital Letter
- Must not have space or slash (/)
- Starting character must not be a number

Assumption:

Input string will not be empty.

Example:

Input 1:

aA1_67

Input 2:

a987 abC012

Output 1:

1

Output 2:

0





Code Solution in C

```
#include <stdio.h>
#include <string.h>

int CheckPassword (char str[], int n)
{
    // At least 4 characters
    if (n < 4)
        return 0;
    // Starting character must not be a number
    if (str[0] - '0' >= 0 && str[0] - '9' <= 9)
        return 0;
    int a = 0, cap = 0, nu = 0;
    while (a < n)
    {
        // Must not have space or slash (/)
        if (str[a] == ' ' || str[a] == '/')
            return 0;
        // Counting capital letters
        if (str[a] >= 'A' && str[a] <= 'Z')
        {
            cap++;
        }
        // Counting numeric digits
        else if (str[a] - '0' >= 0 && str[a] - '9' <= 9)
        {
            nu++;
        }
        // Incrementing for the while loop
        a++;
    }
    // Returns 1 if there are > 0 numeric digits and capital letters
    return cap > 0 && nu > 0;
}

int main ()
{
    char str[100];
    fgets (str, sizeof (str), stdin);
    int len = strlen (str);
    if (str[len - 1] == '\n')
    {
        str[len - 1] = '\0';           // Remove the newline character
        len--;                      // Decrement the length
    }
    int result = CheckPassword (str, len);
    printf ("%d\n", result);
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int CheckPassword (char str[], int n)
{
    //At least 4 characters
    if (n < 4)
        return 0;
    //Starting character must not be a number
    if (str[0] - '0' >= 0 && str[0] - '9' <= 9)
        return 0;
    int a = 0, cap = 0, nu = 0;
    while (a < n)
    {
        //Must not have space or slash (/)
        if (str[a] == ' ' || str[a] == '/')
            return 0;
        //counting capital letters
        if (str[a] >= 65 && str[a] <= 90)
        {
            cap++;
        }
        //counting numeric digit
        else if (str[a] - '0' >= 0 && str[a] - '9' <= 9)
            nu++;
        //incrementing for while loop
        a++;
    }
    // returns 1 if there are > 0 numeric digits and capital letters
    return cap > 0 && nu > 0;
}
int main ()
{
    string s;
    getline (cin, s);
    int len = s.size ();
    char *c = &s[0];
    cout << CheckPassword (c, len);
}
```

Code Solution in JAVA

```
import java.util.*;
class Solution
{
    public static int checkPassword (String str, int n)
    {
        if (n < 4)
            return 0;
        if (str.charAt (0) >= '0' && str.charAt (0) <= '9')
            return 0;
        int num = 0, cap = 0;
        for (int i = 0; i < n; i++)
        {
            if (str.charAt (i) == ' ' || str.charAt (i) == '/')
                return 0;
            if (str.charAt (i) >= 'A' && str.charAt (i) <= 'Z')
                cap++;
            if (str.charAt (i) >= '0' && str.charAt (i) <= '9')
                num++;
        }
        if (cap > 0 && num > 0)
            return 1;
        else
            return 0;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        String str = sc.next ();
        System.out.println (checkPassword (str, str.length ()));
    }
}
```

Code Solution in PYTHON

```
def CheckPassword(s,n):
    if n<4:
        return 0
    if s[0].isdigit():
        return 0
    cap=0
    nu=0
    for i in range(n):
        if s[i]==' ' or s[i]=='/':
            return 0
        if s[i]>='A' and s[i]<='Z':
            cap+=1
        elif s[i].isdigit():
            nu+=1

    if cap>0 and nu>0:
        return 1
    else:
        return 0

s=input()
a=len(s)
print(CheckPassword(s,a))
```





9. PROGRAM 9

You are given a function,

```
int findCount(int arr[], int length, int num, int diff);
```

The function accepts an integer array ‘arr’, its length and two integer variables ‘num’ and ‘diff’. Implement this function to find and return the number of elements of ‘arr’ having an absolute difference of less than or equal to ‘diff’ with ‘num’.

Note: In case there is no element in ‘arr’ whose absolute difference with ‘num’ is less than or equal to ‘diff’, return -1.

Example:

Input:

- arr: 12 3 14 56 77 13
- num: 13
- diff: 2

Output:

3

Explanation:

Elements of ‘arr’ having absolute difference of less than or equal to ‘diff’ i.e. 2 with ‘num’ i.e. 13 are 12, 13 and 14.





Code Solution in C

```
#include <stdio.h>
#include <stdlib.h>

int findCount (int n, int arr[], int num, int diff)
{
    int count = 0;
    for (int i = 0; i < n; ++i)
    {
        if (abs (arr[i] - num) <= diff)
        {
            count++;
        }
    }
    return count > 0 ? count : -1;
}

int main ()
{
    int n;
    scanf ("%d", &n);
    int arr[n];
    for (int i = 0; i < n; ++i)
    {
        scanf ("%d", &arr[i]);
    }
    int num;
    scanf ("%d", &num);
    int diff;
    scanf ("%d", &diff);
    printf ("%d\n", findCount (n, arr, num, diff));
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int findCount (int n, int arr[], int num, int diff)
{
    int count = 0;
    for (int i = 0; i < n; ++i)
    {
        if (abs (arr[i] - num) <= diff)
        {
            count++;
        }
    }
    return count > 0 ? count : -1;
}
int main ()
{
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n; ++i)
    {
        cin >> arr[i];
    }
    int num;
    cin >> num;
    int diff;
    cin >> diff;
    cout << findCount (n, arr, num, diff);
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int findCount (int arr[], int length, int num, int diff)
    {
        int count = 0;
        for (int i = 0; i < length; i++)
        {
            if (Math.abs (num - arr[i]) <= diff)
                count++;
        }
        return count > 0 ? count : -1;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int n = sc.nextInt ();
        int arr[] = new int[n];
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt ();
        int num = sc.nextInt ();
        int diff = sc.nextInt ();

        System.out.println (findCount (arr, n, num, diff));
    }
}
```

Code Solution in PYTHON

```
def findCount(n, arr, num, diff):
    count=0
    for i in range(n):
        if(abs(arr[i]-num)<=diff):
            count+=1
    if count:
        return count
    return 0
n=int(input())
arr=list(map(int,input().split()))
num=int(input())
diff=int(input())
print(findCount(n, arr, num, diff))
```





10. PROGRAM 10

Implement the following Function

```
def differenceofSum(n, m)
```

The function accepts two integers n, m as arguments Find the sum of all numbers in range from 1 to m(both inclusive) that are not divisible by n. Return difference between sum of integers not divisible by n with sum of numbers divisible by n.

Assumption:

- $n > 0$ and $m > 0$
- Sum lies between integral range

Example

Input

n:4

m:20

Output

90

Explanation

- Sum of numbers divisible by 4 are $4 + 8 + 12 + 16 + 20 = 60$
- Sum of numbers not divisible by 4 are $1 + 2 + 3 + 5 + 6 + 7 + 9 + 10 + 11 + 13 + 14 + 15 + 17 + 18 + 19 = 150$
- Difference $150 - 60 = 90$

Sample Input

n:3

m:10

Sample Output

19





Code Solution in C

```
#include<stdio.h>
int differenceofSum (int n, int m)
{
    int i, sum1 = 0, sum2 = 0;
    for (i = 1; i <= m; i++)
    {
        if (i % n == 0)
        {
            sum1 = sum1 + i;
        }
        else
        {
            sum2 = sum2 + i;
        }
    }
    if (sum2 > sum1)
        return sum2 - sum1;
    else
        return sum1 - sum2;
}
int main ()
{
    int n, m;
    int result;
    scanf ("%d", &n);
    scanf ("%d", &m);
    result = differenceofSum (n, m);
    printf ("%d", result);
    return 0;
}
```

Code Solution in C++

```
#include <iostream>
using namespace std;

int differenceofSum (int n, int m)
{
    int i, sum1 = 0, sum2 = 0;
    for (i = 1; i <= m; i++) { if (i % n == 0) { sum1 = sum1 + i; } else { sum2 = sum2 + i; } } if (sum2 > sum1)
        return sum2 - sum1;
    else
        return sum1 - sum2;
}

int main ()
{
    int n, m;
    int result;
    cin >> n;
    cin >> m;
    result = differenceofSum (n, m);
    cout << result;
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Solution
{
    public static int differenceOfSum (int m, int n)
    {
        int sum1 = 0, sum2 = 0;
        for (int i = 1; i <= m; i++)
        {
            if (i % n == 0)
                sum1 = sum1 + i;
            else
                sum2 = sum2 + i;
        }
        return Math.abs (sum1 - sum2);
    }

    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int n = sc.nextInt ();
        int m = sc.nextInt ();
        System.out.println (differenceOfSum (m, n));
    }
}
```

Code Solution in PYTHON

```
n = int(input())
m = int(input())
sum1 = 0
sum2 = 0
for i in range(1,m+1):
    if i % n == 0:
        sum1+=i
    else:
        sum2+=i
print(abs(sum2-sum1))
```





11. PROGRAM 11

You are required to implement the following Function

```
def LargeSmallSum(arr)
```

The function accepts an integers arr of size 'length' as its arguments you are required to return the sum of second largest element from the even positions and second smallest from the odd position of given 'arr'

Assumption:

- All array elements are unique
- Treat the 0th position as even

NOTE

- Return 0 if array is empty
- Return 0, if array length is 3 or less than 3

Example

Input

arr:3 2 1 7 5 4

Output

7

Explanation

- Second largest among even position elements(1 3 5) is 3
- Second smallest among odd position element is 4
- Thus output is $3+4 = 7$

Sample Input

arr:1 8 0 2 3 5 6

Sample Output

8





Code Solution in C

```
#include <stdio.h>

int LargeSmallSum (int arr[], int length)
{
    if (length <= 3)
    {
        return 0;
    }

    int evenPos[100], oddPos[100];
    int evenCount = 0, oddCount = 0;

    for (int i = 0; i < length; i++)
    {
        if (i % 2 == 0)
        {
            evenPos[evenCount++] = arr[i];
        }
        else
        {
            oddPos[oddCount++] = arr[i];
        }
    }

    // Sort the even position array in descending order
    for (int i = 0; i < evenCount - 1; i++)
    {
        for (int j = 0; j < evenCount - i - 1; j++)
        {
            if (evenPos[j] < evenPos[j + 1])
            {
                int temp = evenPos[j];
                evenPos[j] = evenPos[j + 1];
                evenPos[j + 1] = temp;
            }
        }
    }

    // Sort the odd position array in ascending order
    for (int i = 0; i < oddCount - 1; i++)
    {
        for (int j = 0; j < oddCount - i - 1; j++)
        {
            if (oddPos[j] > oddPos[j + 1])
            {
                int temp = oddPos[j];
                oddPos[j] = oddPos[j + 1];
                oddPos[j + 1] = temp;
            }
        }
    }

    return evenPos[1] + oddPos[1];
}

int main ()
{
    int arr[] = { 3, 2, 1, 7, 5, 4 };
    int length = sizeof (arr) / sizeof (arr[0]);

    int result = LargeSmallSum (arr, length);
    printf ("%d\n", result);

    return 0;
}
```

Code Solution in C++

```
#include <iostream>
#include <algorithm>
#include <vector>

using namespace std;

int LargeSmallSum (vector < int >&arr)
{
    int length = arr.size ();

    if (length <= 3)
    {
        return 0;
    }

    vector < int >evenPos;
    vector < int >oddPos;

    for (int i = 0; i < length; i++)
    {
        if (i % 2 == 0)
        {
            evenPos.push_back (arr[i]);
        }
        else
        {
            oddPos.push_back (arr[i]);
        }
    }

    // Sort the even position array in descending order
    sort (evenPos.rbegin (), evenPos.rend ());

    // Sort the odd position array in ascending order
    sort (oddPos.begin (), oddPos.end ());

    return evenPos[1] + oddPos[1];
}

int main ()
{
    vector < int >arr = { 3, 2, 1, 7, 5, 4 };

    int result = LargeSmallSum (arr);
    cout << result << endl;

    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int largeSmallSum (int[]arr, int n)
    {
        if (n <= 3)
            return 0;
        ArrayList < Integer > even = new ArrayList < Integer > ();
        ArrayList < Integer > odd = new ArrayList < Integer > ();
        even.add (arr[0]);

        for (int i = 1; i < arr.length; i++)
        {
            if (i % 2 == 0)
                even.add (arr[i]);
            else
                odd.add (arr[i]);
        }
        Collections.sort (even);
        Collections.sort (odd);

        return even.get (even.size () - 2) + odd.get (odd.size () - 2);
    }

    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int n = sc.nextInt ();
        int arr[] = new int[n];
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt ();

        System.out.println (largeSmallSum (arr, n));
    }
}
```

Code Solution in PYTHON

```
length = int(input())
arr = list(map(int, input().split()))
even_arr = []
odd_arr = []
for i in range(length):
    if i % 2 == 0:
        even_arr.append(arr[i])
    else:
        odd_arr.append(arr[i])
even_arr = sorted(even_arr)
odd_arr = sorted(odd_arr)
print(even_arr[len(even_arr)-2] + odd_arr[len(odd_arr)-2])
```





12. PROGRAM 12

Implement the following Function

```
def ProductSmallestPair(sum, arr)
```

The function accepts an integer sum and an integer array arr of size n. Implement the function to find the pair, (arr[j], arr[k]) where j!=k, Such that arr[j] and arr[k] are the least two elements of array ($arr[j] + arr[k] \leq sum$) and return the product of element of this pair

NOTE

- Return -1 if array is empty or if $n < 2$
- Return 0, if no such pairs found
- All computed values lie within integer range

Example

Input

sum:9

size of Arr = 7

Arr:5 2 4 3 9 7 1

Output

2

Explanation

Pair of least two element is (2, 1) $2 + 1 = 3 < 9$, Product of (2, 1) $2 * 1 = 2$. Thus, output is 2

Sample Input

sum:4

size of Arr = 6

Arr:9 8 3 -7 3 9

Sample Output

-21





Code Solution in C

```
#include<stdio.h>
int productSmallestPair (int *array, int n, int sum)
{
    int answer, temp, i, j, check;
    if (n < 2)
    {
        answer = -1;
    }
    else
    {
        for (i = 0; i < n; i++) //sorting of array
        {
            for (j = i + 1; j < n; j++)
            {
                if (array[i] > array[j])
                {
                    temp = array[i];
                    array[i] = array[j];
                    array[j] = temp;
                }
            }
        }
        check = array[0] + array[1];
        if (check <= sum)
        {
            answer = array[0] * array[1];
        }
        else
        {
            answer = 0;
        }
    }
    return answer;
}
int main ()
{
    int n, sum, result, i;
    scanf ("%d", &sum);
    scanf ("%d", &n);
    int array[n];
    for (i = 0; i < n; i++)
    {
        scanf ("%d", &array[i]);
    }
    result = productSmallestPair (array, n, sum);
    printf ("%d", result);
    return 0;
}
```

Code Solution in C++

```
#include <iostream>
#include <algorithm>

int productSmallestPair (int *array, int n, int sum)
{
    int answer, temp, i, j, check;
    if (n < 2)
    {
        answer = -1;
    }
    else
    {
        for (i = 0; i < n; i++) // sorting of array
        {
            for (j = i + 1; j < n; j++)
            {
                if (array[i] > array[j])
                {
                    temp = array[i];
                    array[i] = array[j];
                    array[j] = temp;
                }
            }
        }
        check = array[0] + array[1];
        if (check <= sum)
        {
            answer = array[0] * array[1];
        }
        else
        {
            answer = 0;
        }
    }
    return answer;
}

int main ()
{
    int n, sum, result, i;
    std::cin >> sum;
    std::cin >> n;
    int *array = new int[n];
    for (i = 0; i < n; i++)
    {
        std::cin >> array[i];
    }
    result = productSmallestPair (array, n, sum);
    std::cout << result;
    delete []array;
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int productSmallestPair (int arr[], int n, int sum)
    {
        if (n < 2)
            return -1;
        int ans, temp, check;
        for (int i = 0; i < n; i++)
        {
            for (int j = i + 1; j < n; j++)
            {
                if (arr[i] > arr[j])
                {
                    temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
        check = arr[0] + arr[1];
        if (check <= sum)
            return arr[0] * arr[1];
        else
            return 0;
    }

    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int sum = sc.nextInt ();
        int n = sc.nextInt ();
        int arr[] = new int[n];

        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt ();
        System.out.println (productSmallestPair (arr, n, sum));
    }
}
```

Code Solution in PYTHON

```
n = int(input())
sum1 = int(input())
arr = list(map(int, input().split()))
if n < 2:
    print('-1')
arr = sorted(arr)
for i in range(n-1):
    if arr[i] + arr[i+1] < sum1:
        print(arr[i] * arr[i+1])
        break
else:
    print('0')
```



13. PROGRAM 13

N-base notation is a system for writing numbers that uses only n different symbols, This symbols are the first n symbols from the given notation list(Including the symbol for 0) Decimal to n base notation are (0:0, 1:1, 2:2, 3:3, 4:4, 5:5, 6:6, 7:7, 8:8, 9:9, 10:A,11:B and so on up to 35:Z)

Implement the following function

Char* DectoNBase (int n, int num):

The function accept positive integer n and num Implement the function to calculate the n-base equivalent of num and return the same as a string

Steps:

1. Divide the decimal number by n, Treat the division as the integer division
2. Write the remainder (in n-base notation)
3. Divide the quotient again by n, Treat the division as integer division
4. Repeat step 2 and 3 until the quotient is 0
5. The n-base value is the sequence of the remainders from last to first

Assumption:

$1 < n \leq 36$

Example

Input

n: 12

num: 718

Output

4BA

Explanation

num	Divisor	quotient	remainder
718	12	59	10(A)
59	12	4	11(B)
4	12	0	4(4)

Sample Input

n: 21

num: 5678

Sample Output

CI8





Code Solution in C

```
#include <stdio.h>
#include <stdlib.h>

char *decitoNBase (int n, int num)
{
    char *res = NULL;
    int quotient = num / n;
    int *rem = malloc (sizeof (int));
    int remSize = 1;
    rem[0] = num % n;

    while (quotient != 0)
    {
        remSize++;
        int *temp = realloc (rem, remSize * sizeof (int));
        if (temp == NULL)
        {
            free (rem);
            return NULL;
        }
        rem = temp;
        rem[remSize - 1] = quotient % n;
        quotient = quotient / n;
    }

    int resSize = remSize + 1;
    res = malloc (resSize * sizeof (char));
    if (res == NULL)
    {
        free (rem);
        return NULL;
    }

    int i, j = 0;
    for (i = remSize - 1; i >= 0; i--)
    {
        if (rem[i] > 9)
        {
            res[j] = (char) (rem[i] - 9 + 64);
        }
        else
        {
            res[j] = (char) (rem[i] + '0');
        }
        j++;
    }
    res[j] = '\0';
    free (rem);
    return res;
}

int main ()
{
    int n, num;
    scanf ("%d %d", &n, &num);
    char *result = decitoNBase (n, num);
    printf ("%s\n", result);
    free (result);
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
string decitoNBase (int n, int num)
{
    string res = "";
    int quotient = num / n;
    vector < int >rem;
    rem.push_back (num % n);
    while (quotient != 0)
    {
        rem.push_back (quotient % n);
        quotient = quotient / n;
    }
    for (int i = 0; i < rem.size (); i++)
    {
        if (rem[i] > 9)
        {
            res = (char) (rem[i] - 9 + 64) + res;
        }
        else
            res = to_string (rem[i]) + res;
    }
    return res;
}
int main ()
{
    int n, num;
    cin >> n >> num;
    cout << decitoNBase (n, num);
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static String dectoNBase (int n, int num)
    {
        String res = "";
        int quotient = num / n;
        ArrayList < Integer > rem = new ArrayList < Integer > ();
        rem.add (num % n);
        while (quotient != 0)
        {
            rem.add (quotient % n);
            quotient = quotient / n;
        }
        for (int i = 0; i < rem.size (); i++)
        {
            if (rem.get (i) > 9)
            {
                res = (char) (rem.get (i) - 9 + 64) + res;
            }
            else
                res = rem.get (i) + res;
        }
        return res;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int n = sc.nextInt ();
        int num = sc.nextInt ();
        System.out.println (dectoNBase (n, num));
    }
}
```

Code Solution in PYTHON

```
n = int(input())
num = int(input())
reminder = []
quotient = num // n
reminder.append(num%n)
while quotient != 0:
    reminder.append(quotient%n)
    quotient = quotient // n
reminder = reminder[::-1]
equivalent = ''
for i in reminder:
    if i > 9:
        a = i - 9
        a = 64 + a
        equivalent+=chr(a)
    else:
        equivalent+=str(i)
print(equivalent)
```





14. PROGRAM 14

Implement the following functions

```
char*MoveHyphen(char str[],int n);
```

The function accepts a string “str” of length ‘n’, that contains alphabets and hyphens (-).

Implement the function to move all hyphens(-) in the string to the front of the given string.

NOTE: Return null if str is null.

Example:

Input:

str.Move-Hyphens-to-Front

Output:

—MoveHyphenstoFront

Explanation:

The string “Move-Hyphens -to-front” has 3 hyphens (-), which are moved to the front of the string, this output is “— MoveHyphen”

Sample Input

Str: String-Compare

Sample Output

StringCompare



Code Solution in C

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char * MoveHyphen (char *s, int n)
{
    int count = 0;
    int i = 0;
    while (i < n)
    {
        if (s[i] == '-')
        {
            count++;
            memmove (&s[i], &s[i + 1], n - i);
            n--;
        }
        else
        {
            i++;
        }
    }

    char *res = malloc ((n + count + 1) * sizeof (char));
    if (res == NULL)
    {
        return NULL;
    }

    for (i = 0; i < count; i++)
    {
        res[i] = '-';
    }
    for (i = count; i < count + n; i++)
    {
        res[i] = s[i - count];
    }
    res[count + n] = '\0';

    return res;
}

int main ()
{
    char s[100];
    scanf ("%s", s);
    int n = strlen (s);
    char *result = MoveHyphen (s, n);
    printf ("%s\n", result);
    free (result);
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
string MoveHyphen (string s, int n)
{
    int count = 0;
    for (int i = 0; i < n;)
    {
        if (s[i] == '-')
        {
            count++;
            s.erase (i, 1);
        }
        else
        {
            i++;
        }
    }
    while (count--)
    {
        s = '-' + s;
    }
    return s;
}

int main ()
{
    string s;
    cin >> s;
    int n = s.size ();
    cout << MoveHyphen (s, n);
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Solution
{
    public static String moveHyphen (String str, int n)
    {
        String res = "";
        for (int i = 0; i < n; i++)
        {
            if (str.charAt (i) == '-')
                res = str.charAt (i) + res;
            else
                res = res + str.charAt (i);
        }
        return res;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        String str = sc.next ();

        System.out.println (moveHyphen (str, str.length ()));
    }
}
```

Code Solution in PYTHON

```
inp = input()
count = 0
final = ""
for i in inp:
    if i == '-':
        count+=1
    else:
        final+=i
print("-"*count,final)
```



15. PROGRAM 15

Problem Statement

A carry is a digit that is transferred to left if sum of digits exceeds 9 while adding two numbers from right-to-left one digit at a time

You are required to implement the following function.

```
Int NumberOfCarries(int num1 , int num2);
```

The function accepts two numbers ‘num1’ and ‘num2’ as its arguments. You are required to calculate and return the total number of carries generated while adding digits of two numbers ‘num1’ and ‘num2’.

Assumption: num1, num2 ≥ 0

Example:

- **Input**
 - Num 1: 451
 - Num 2: 349
- **Output**
 - 2

Explanation:

Adding ‘num 1’ and ‘num 2’ right-to-left results in 2 carries since (1+9) is 10. 1 is carried and (5+4=1) is 10, again 1 is carried. Hence 2 is returned.

Sample Input

Num 1: 23

Num 2: 563

Sample Output

0





Code Solution in C

```
#include<stdio.h>
int numberofCarries (int num1, int num2)
{
    int carry = 0, sum, p, q, count = 0;
    while ((num1 != 0) && (num2 != 0))
    {
        p = num1 % 10;
        q = num2 % 10;
        sum = carry + p + q;
        if (sum > 9)
        {
            carry = 1;
            count++;
        }
        else
        {
            carry = 0;
        }
        num1 = num1 / 10;
        num2 = num2 / 10;
    }
    while (num1 != 0)
    {
        p = num1 % 10;
        sum = carry + p;
        if (sum > 9)
        {
            carry = 1;
            count++;
        }
        else
        {
            carry = 0;
        }
        num1 = num1 / 10;
    }
    while (num2 != 0)
    {
        q = num2 % 10;
        sum = carry + q;
        if (sum > 9)
        {
            carry = 1;
            count++;
        }
        else
        {
            carry = 0;
        }
        num2 = num2 / 10;
    }
    return count;
}
int main ()
{
    int x, y, a;
    scanf ("%d", &x);
    scanf ("%d", &y);
    a = numberofCarries (x, y);
    printf ("%d", a);
    return 0;
}
```

Code Solution in C++

```
#include <iostream>
using namespace std;
int numberofCarries (int num1, int num2)
{
    int carry = 0, sum, p, q, count = 0;
    while ((num1 != 0) && (num2 != 0))
    {
        p = num1 % 10;
        q = num2 % 10;
        sum = carry + p + q;
        if (sum > 9)
        {
            carry = 1;
            count++;
        }
        else
        {
            carry = 0;
        }
        num1 = num1 / 10;
        num2 = num2 / 10;
    }
    while (num1 != 0)
    {
        p = num1 % 10;
        sum = carry + p;
        if (sum > 9)
        {
            carry = 1;
            count++;
        }
        else
        {
            carry = 0;
        }
        num1 = num1 / 10;
    }
    while (num2 != 0)
    {
        q = num2 % 10;
        sum = carry + q;
        if (sum > 9)
        {
            carry = 1;
            count++;
        }
        else
        {
            carry = 0;
        }
        num2 = num2 / 10;
    }
    return count;
}
int main ()
{
    int x, y, a;
    cin >> x >> y;
    a = numberofCarries (x, y);
    cout << a;
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int numberofCarries (int num1, int num2)
    {
        int count = 0;
        int temp1 = num1, temp2 = num2;
        int rem = 0;
        while (temp1 != 0 && temp2 != 0)
        {
            int d1 = temp1 % 10, d2 = temp2 % 10;
            if ((d1 + d2 + rem) > 9)
            {
                count++;
                int sum = d1 + d2 + rem;
                sum = sum / 10;
                rem = sum;
            }
            temp1 = temp1 / 10;
            temp2 = temp2 / 10;
        }
        while (temp1!=0)
        {
            int d1=temp1%10;
            if((d1+rem)>9)
            {
                count++;
                int sum=d1+rem;
                sum=sum/10;
                rem=sum;
            }
            temp1=temp1/10;
        }
        while(temp2!=0)
        {
            int d2=temp2%10;
            if((d2+rem)>9)
            {
                count++;
                int sum=d2+rem;
                sum=sum/10;
                rem=sum;
            }
            temp2=temp2/10;
        }
        return count;
    }
    public static void main (String[] args)
    {
        Scanner sc = new Scanner (System.in);
        int num1 = sc.nextInt ();
        int num2 = sc.nextInt ();
        System.out.println (numberofCarries (num1, num2));
    }
}
```

Code Solution in PYTHON

```
def NumberofCarries(n1,n2):
    count=0
    carry = 0
    if len(n1) <= len(n2):
        l= len(n1)-1
    else:
        l = len(n2)-1
    for i in range(l+1):
        temp = int(n1[l-i])+int(n2[l-i])+carry
        if len(str(temp))>1:
            count+=1
            carry = 1
        else:
            carry = 0
    return count+carry
n1=input()
n2=input()
print(NumberOfCarries(n1,n2))
```



16. PROGRAM 16

Problem Statement

You are given a function,

```
Void *ReplaceCharacter(Char str[], int n, char ch1, char ch2);
```

The function accepts a string ‘str’ of length n and two characters ‘ch1’ and ‘ch2’ as its arguments . Implement the function to modify and return the string ‘str’ in such a way that all occurrences of ‘ch1’ in original string are replaced by ‘ch2’ and all occurrences of ‘ch2’ in original string are replaced by ‘ch1’.

Assumption: String Contains only lower-case alphabetical letters.

Note:

- Return null if string is null.
- If both characters are not present in string or both of them are same , then return the string unchanged.

Example:

- **Input:**
 - Str: apples
 - ch1:a
 - ch2:p
- **Output:**
 - paales

Explanation:

‘A’ in the original string is replaced with ‘p’ and ‘p’ in original string is replaced with ‘a’, thus output is paales.



Code Solution in C

```
#include<stdio.h>
#include<string.h>
void *ReplaceCharacter (char str[], int n, char ch1, char ch2)
{
    int i;
    for (i = 0; i < n; i++)
    {
        if (str[i] == ch1)
        {
            str[i] = ch2;
        }
        else if (str[i] == ch2)
        {
            str[i] = ch1;
        }
    }
    printf ("%s", str);
}
int main ()
{
    char a[100];
    char b, c;
    int len;
    scanf ("%s", a);
    scanf ("%s", &b);
    scanf ("%s", &c);
    len = strlen (a);
    ReplaceCharacter (a, len, b, c);
    return 0;
}
```

Code Solution in C++

```
#include <iostream>
#include <cstring>
using namespace std;
void ReplaceCharacter (char str[], int n, char ch1, char ch2)
{
    for (int i = 0; i < n; i++)
    {
        if (str[i] == ch1)
        {
            str[i] = ch2;
        }
        else if (str[i] == ch2)
        {
            str[i] = ch1;
        }
    }
    cout << str;
}

int main ()
{
    char a[100];
    char b, c;
    int len;

    cin >> a;
    cin >> b;
    cin >> c;

    len = std::strlen (a);
    ReplaceCharacter (a, len, b, c);

    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Solution
{
    public static void replaceChar (String str, char ch1, char ch2)
    {
        String res = "";
        for (int i = 0; i < str.length (); i++)
        {
            if (str.charAt (i) == ch1)
                res = res + ch2;
            else if (str.charAt (i) == ch2)
                res = res + ch1;
            else
                res = res + str.charAt (i);
        }
        System.out.println (res);
    }

    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        String str = sc.next ();
        char ch1 = sc.next ().charAt (0);
        char ch2 = sc.next ().charAt (0);

        replaceChar (str, ch1, ch2);
    }
}
```

Code Solution in PYTHON

```
def swap (user_input, str1, str2):
    result = ""
    if user_input != None:
        result = user_input.replace(str1, str2).replace(str2, str1).replace(str1, str2)
    return result
user_input = input()
str1, str2 = map(str, input().split())
print(swap(user_input, str1, str2))
```





17. PROGRAM 17 - OPERATION CHOICES

Problem Statement

You are required to implement the following function.

Int OperationChoices(int c, int n, int a , int b)

The function accepts 3 positive integers ‘a’ , ‘b’ and ‘c ‘ as its arguments. Implement the function to return.

- (a + b) , if c=1
- (a – b) , if c=2
- (a * b) , if c=3
- (a / b) , if c =4

Assumption: All operations will result in integer output.

Example:

- **Input**
 - c :1
 - a:12
 - b:16
- **Output:**
 - Since ‘c’=1 , (12+16) is performed which is equal to 28 , hence 28 is returned.

Sample Input

c : 2

a : 16

b : 20

Sample Output

-4





Code Solution in C

```
#include<stdio.h>
int operationChoices (int c, int a, int b)
{
    if (c == 1)
    {
        return a + b;
    }
    else if (c == 2)
    {
        return a - b;
    }
    else if (c == 3)
    {
        return a * b;
    }
    else if (c == 4)
    {
        return a / b;
    }
}
int main ()
{
    int x, y, z;
    int result;
    scanf ("%d", &x);
    scanf ("%d", &y);
    scanf ("%d", &z);
    result = operationChoices (x, y, z);
    printf ("%d", result);
    return 0;
}
```

Code Solution in C++

```
#include <iostream>
using namespace std;
int operationChoices(int c, int a, int b)
{
    if (c == 1) {
        return a + b;
    }
    else if (c == 2) {
        return a - b;
    }
    else if (c == 3) {
        return a * b;
    }
    else if (c == 4) {
        return a / b;
    }
    return 0;
}

int main()
{
    int x, y, z;
    int result;
    cin >> x;
    cin >> y;
    cin >> z;
    result = operationChoices(x, y, z);
    cout << result;
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Solution
{
    public static int operationChoices (int c, int a, int b)
    {
        int res = 0;
        switch (c)
        {
            case 1:
                res = a + b;
                break;
            case 2:res = a - b;
                break;
            case 3:res = a * b;
                break;
            case 4:res = a / b;
                break;
        }
        return res;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int c = sc.nextInt ();
        int a = sc.nextInt ();
        int b = sc.nextInt ();

        System.out.println (operationChoices (c, a, b));
    }
}
```

Code Solution in PYTHON

```
def operationChoices(c,a,b):
    if c == 1 :
        return(a+b)
    elif c == 2:
        return(a-b)
    elif c == 3:
        return(a*b)
    else:
        return(a//b)
c,a,b = map(int,input().split())
print(operationChoices(c, a, b))
```



18. PROGRAM 18

Problem Statement

You are given a function,

```
Int MaxExponents (int a , int b);
```

You have to find and return the number between ‘a’ and ‘b’ (range inclusive on both ends)

which has the maximum exponent of 2.

The algorithm to find the number with maximum exponent of 2 between the given range is

1. Loop between ‘a’ and ‘b’. Let the looping variable be ‘i’.
2. Find the exponent (power) of 2 for each ‘i’ and store the number with maximum exponent of 2 so far in a variable , let say ‘max’. Set ‘max’ to ‘i’ only if ‘i’ has more exponent of 2 than ‘max’.
3. Return ‘max’.

Assumption: a < b

Note: If two or more numbers in the range have the same exponents of 2 , return the small number.

Example

- **Input:**
 - 7
 - 12
- **Output:**
 - 8

Explanation:

Exponents of 2 in:

7-0

8-3

9-0

10-1

11-0

12-2

Hence maximum exponent of two is of 8.





Code Solution in C

```
#include <stdio.h>

int count (int n)
{
    int c = 0;
    while (n % 2 == 0 && n != 0)
    {
        c++;
        n = n / 2;
    }
    return c;
}

int maxExponents (int a, int b)
{
    int max = 0, num = 0, ans;
    for (int i = a; i <= b; i++)
    {
        int temp = count (i);
        if (temp > max)
        {
            max = temp;
            num = i;
        }
    }
    return num;
}

int main ()
{
    int a, b;
    scanf ("%d %d", &a, &b);
    printf ("%d", maxExponents (a, b));
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int count (int n)
{
    int c = 0;
    while (n % 2 == 0 && n != 0)
    {
        c++;
        n = n / 2;
    }
    return c;
}
int maxExponents (int a, int b)
{
    int max = 0, num = 0, ans;
    for (int i = a; i <= b; i++) { int temp = count (i); if (temp > max)
    {
        max = temp;
        num = i;
    }
    }
    return num;
}
int main ()
{
    int a, b;
    cin >> a >> b;
    cout << maxExponents (a, b);
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static int count (int i)
    {
        int count = 0;
        while (i % 2 == 0 && i != 0)
        {
            count += 1;
            i = i / 2;
        }
        return count;
    }
    public static int maxExponents (int a, int b)
    {
        int max = 0, num = 0, ans;
        for (int i = a; i <= b; i++)
        {
            int temp = count (i);
            if (temp > max)
            {
                max = temp;
                num = i;
            }
        }
        return num;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int a = sc.nextInt ();
        int b = sc.nextInt ();
        System.out.println (maxExponents(a, b));
    }
}
```

Code Solution in PYTHON

```
def countExponents(i):
    count = 0
    while i%2 == 0 and i != 0:
        count+=1
        i = i//2
    return count
def maxExponents(a, b):
    maximum, number = 0, a
    for i in range(a,b):
        temp = countExponents(i)
        if temp>maximum:
            maximum, number = temp, i
    return number
a, b = map(int,input().split())
print(maxExponents(a, b))
```





19. PROGRAM 19

Problem Statement

You are required to implement the following function:

```
Int Calculate(int m, int n);
```

The function accepts 2 positive integer ‘m’ and ‘n’ as its arguments. You are required to calculate the sum of numbers divisible both by 3 and 5, between ‘m’ and ‘n’ both inclusive and return the same.

Note

$0 < m \leq n$

Example

Input:

m : 12

n : 50

Output

90

Explanation:

The numbers divisible by both 3 and 5, between 12 and 50 both inclusive are {15, 30, 45} and their sum is 90.

Sample Input

m : 100

n : 160

Sample Output

510





Code Solution in C

```
#include<stdio.h>
int Calculate (int, int);
int main ()
{
    int m, n, result;
    printf ("Enter the value of m : ");
    scanf ("%d", &m);
    printf ("Enter the value of n : ");
    scanf ("%d", &n);
    result = Calculate (n, m);
    printf ("%d", result);
    return 0;
}
int Calculate (int n, int m)
{
    int i, sum = 0;
    for (i = m; i <= n; i++)
    {
        if ((i % 3 == 0) && (i % 5 == 0))
        {
            sum = sum + i;
        }
    }
    return sum;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int Calculate (int, int);
int main ()
{
    int m, n, result;
    cout << "Enter the value of m :"; cin >> m;
    cout << "Enter the value of n :"; cin >> n;
    result = Calculate (n, m);
    cout << result;
    return 0;
}
int Calculate (int n, int m)
{
    int i, sum = 0;
    for (i = m; i <= n; i++)
    {
        if ((i % 3 == 0) && (i % 5 == 0))
        {
            sum = sum + i;
        }
    }
    return sum;
}
```

Code Solution in JAVA

```
import java.util.Scanner;
public class Main
{
    int Calculate (int m, int n)
    {
        int sum = 0;
        for (int i = m; i <= n; i++)
        {
            if ((i % 3 == 0) && (i % 5 == 0))
                sum = sum + i;
        }
        return sum;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter the value of m and n");
        int m = sc.nextInt ();
        int n = sc.nextInt ();
        Main q = new Main ();
        int result = q.Calculate (m, n);
        System.out.println (result);
    }
}
```

Code Solution in PYTHON

```
def Calculate(m, n):
    sum = 0
    for i in range(m, n+1):
        if (i % 3 == 0) and (i % 5 == 0):
            sum = sum + i
    return sum

m = int(input("Enter the value of m: "))
n = int(input("Enter the value of n: "))
result = Calculate(m, n)
print(result)
```



20. PROGRAM 20

Problem Statement

You are required to input the size of the matrix then the elements of matrix, then you have to divide the main matrix in two sub matrices (even and odd) in such a way that element at 0 index will be considered as even and element at 1st index will be considered as odd and so on. Then, you have to sort the even and odd matrices in ascending order and then print the sum of the second largest number from both the matrices.

Example

- enter the size of array : 5
- enter element at 0 index : 3
- enter element at 1 index : 4
- enter element at 2 index : 1
- enter element at 3 index : 7
- enter element at 4 index : 9

Sorted even array: 1 3 9

Sorted odd array: 4 7



Code Solution in C

```

#include <stdio.h>
main()
{
    int arr[100];
    int length, i, j, oddlen, evenlen, temp;
    int odd[50], even[50];
    printf("Enter the length of the array: ");
    scanf("%d", &length);

    for (i = 0; i < length; i++) {
        printf("Enter element at index %d: ", i);
        scanf("%d", &arr[i]);
    }

    if (length % 2 == 0) {
        oddlen = length / 2;
        evenlen = length / 2;
    } else {
        oddlen = length / 2;
        evenlen = (length / 2) + 1;
    }

    for (i = 0; i < length; i++) {
        if (i % 2 == 0) {
            even[i / 2] = arr[i];
        } else {
            odd[i / 2] = arr[i];
        }
    }

    for (i = 0; i < evenlen - 1; i++) {
        for (j = i + 1; j < evenlen; j++) {
            if (even[i] > even[j]) {
                temp = even[i];
                even[i] = even[j];
                even[j] = temp;
            }
        }
    }

    for (i = 0; i < oddlen - 1; i++) {
        for (j = i + 1; j < oddlen; j++) {
            if (odd[i] > odd[j]) {
                temp = odd[i];
                odd[i] = odd[j];
                odd[j] = temp;
            }
        }
    }

    printf("\nSorted even array: ");
    for (i = 0; i < evenlen; i++) {
        printf("%d ", even[i]);
    }
    printf("\n");

    printf("Sorted odd array: ");
    for (i = 0; i < oddlen; i++) {
        printf("%d ", odd[i]);
    }
    printf("\n");

    printf("%d", even[evenlen - 2] + odd[oddlen - 2]);
    return 0;
}

```

Code Solution in C++

```

#include <iostream>
using namespace std;
int main ()
{
    int arr[100];
    int length, i, j, oddlen, evenlen, temp, c, d;
    int odd[50], even[50];
    cout << "Enter the length of array : ";
    cin >> length;
    for (i = 0; i < length; i++)
    {
        cout << "Enter element at " << i << " index : ";
        cin >> arr[i];
    }
    if (length % 2 == 0)
    {
        oddlen = length / 2;
        evenlen = length / 2;
    }
    else
    {
        oddlen = length / 2;
        evenlen = (length / 2) + 1;
    }
    for (i = 0; i < length; i++) // separation of even and odd array
    {
        if (i % 2 == 0)
        {
            even[i / 2] = arr[i];
        }
        else
        {
            odd[i / 2] = arr[i];
        }
    }
    for (i = 0; i < evenlen - 1) // sorting of even array
    {
        for (j = i + 1; j < evenlen; j++)
        {
            if (even[i] > even[j])
            {
                temp = even[i];
                even[i] = even[j];
                even[j] = temp;
            }
        }
    }
    for (i = 0; i < oddlen - 1) // sorting of odd array
    {
        for (j = i + 1; j < oddlen; j++)
        {
            if (odd[i] > odd[j])
            {
                temp = odd[i];
                odd[i] = odd[j];
                odd[j] = temp;
            }
        }
    }
    cout << "\nSorted even array : "; // printing even array
    for (i = 0; i < evenlen; i++)
    {
        cout << even[i] << " ";
    }
    cout << "\n";
    cout << "\nSorted odd array : "; // printing odd array
    for (i = 0; i < oddlen; i++)
    {
        cout << odd[i] << " ";
    }
    cout << endl;
    cout << even[evenlen - 2] + odd[oddlen - 2]; // printing final result
}

```

Code Solution in JAVA

```

import java.util.Scanner;
import java.util.ArrayList;
import java.util.Collections;
public class Main
{
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        System.out.print ("Enter size of array: ");
        int arrsize = sc.nextInt ();
        int[] main = new int[arrsize];
        ArrayList< Integer > even = new ArrayList ();
        ArrayList< Integer > odd = new ArrayList ();
        System.out.println ("Enter " + arrsize + " Elements");
        for (int i = 0; i < arrsize; i++)
            main[i] = sc.nextInt ();
        for (int i = 0; i < arrsize; i++)
        {
            if (i % 2 == 0)
                even.add (main[i]);
            else
                odd.add (main[i]);
        }
        Collections.sort (even);
        Collections.sort (odd);
        System.out.println ("Sorted even array ");
        for (int e:even)
            System.out.print (e + " ");
        System.out.println ();
        System.out.println ("sorted odd array ");
        for (int e:odd)
            System.out.print (e + " ");
        System.out.println ();
        int evensec = even.get (even.size () - 2);
        int oddsec = odd.get (odd.size () - 2);
        System.out.println ("Second Largest Element in Even List is:" + evensec);
        System.out.println ("Second Largest Element in Odd List is:" + oddsec);
        System.
        out.println ("Sum Of Second Largest Element Of Odd and Even List:" +
                    (evensec + oddsec));
    }
}

```

Code Solution in PYTHON

```

array = []
evenArr = []
oddArr = []
n = int(input("Enter the size of the array:"))
for i in range(0,n):
    number = int(input("Enter Element at {} index:".format(i)))
    array.append(number)
    if i % 2 == 0:
        evenArr.append(array[i])
    else:
        oddArr.append(array[i])
evenArr = sorted(evenArr)
print("Sorted Even Array:", evenArr[0:len(evenArr)])
oddArr = sorted(oddArr)
print("Sorted Odd Array:", oddArr[0:len(oddArr)])
print(evenArr[1] + oddArr[1])
evenArr = sorted(evenArr)
print("Sorted Even Array:", evenArr[0:len(evenArr)])
oddArr = sorted(oddArr)
print("Sorted Odd Array:", oddArr[0:len(oddArr)])
print(evenArr[1] + oddArr[1])

```





21. PROGRAM 21

Instructions: You are required to write the code. You can click on compile and run anytime to check compilation/execution status. The code should be logically/syntactically correct.

Problem: Write a program in C to display the table of a number and print the sum of all the multiples in it.

Test Cases:

Test Case 1:

Input:

5

Expected Result Value:

5, 10, 15, 20, 25, 30, 35, 40, 45, 50

275

Test Case 2:

Input:

12

Expected Result Value:

12, 24, 36, 48, 60, 72, 84, 96, 108, 120

660





Code Solution in C

```
#include<stdio.h>
int main ()
{
    int n, i, value = 0, sum = 0;
    printf ("Enter the number for which you want to know the table : ");
    scanf ("%d", &n);
    for (i = 1; i <= 10; ++i)
    {
        value = n * i;
        printf ("%d ", value);
        sum = sum + value;
    }
    printf ("\nsum is %d", sum);
    return 0;
}
```

Code Solution in C++

```
#include<iostream>
using namespace std;
int main ()
{
    int n, i, value = 0, sum = 0;
    cout << "Enter the number for which you want to know the table: ";
    cin >> n;
    for (i = 1; i <= 10; ++i)
    {
        value = n * i;
        cout << value << " ";
        sum = sum + value;
    }
    cout << "\nsum is " << sum;
    return 0;
}
```

Code Solution in JAVA

```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.Collections;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Any No:");
        int no = sc.nextInt();
        int sum=0,value=1;
        for(int i=1 ; i<=10 ; i++) {
            value = no*i;
            System.out.print(value+" ");
            sum=sum+value;
        }
        System.out.println("\nSum is : "+sum);
    }
}
```

Code Solution in PYTHON

```
table_number = int(input())
sum = 0
for i in range(1, 11):
    value = table_number * i
    print(value, end=" ")
    sum = sum + value
print()
print(sum)
```





22. PROGRAM 22 - INTERMEDIATE PALINDROME NUMBERS

Instructions: You are required to write the code. You can click on compile and run anytime to check compilation/execution status. The code should be logically/syntactically correct.

Question: Write a program such that it takes a lower limit and upper limit as inputs and print all the intermediate palindrome numbers.

Test Cases:

TestCase 1:

Input :

10 , 80

Expected Result:

11 , 22 , 33 , 44 , 55 , 66 , 77.

Test Case 2:

Input:

100, 200

Expected Result:

101 , 111 , 121 , 131 , 141 , 151 , 161 , 171 , 181 , 191.





Code Solution in C

```
#include<stdio.h>
int main ()
{
    int i, n, reverse, d, f, l;
    printf ("enter the starting \n");
    scanf ("%d", &f);
    printf ("enter the ending\n");
    scanf ("%d", &l);
    for (i = f; i <= l; i++)
    {
        reverse = 0;
        n = i;
        while (n != 0)
        {
            d = n % 10;
            reverse = reverse * 10 + d;
            n = n / 10;
        }
        if (i == reverse)
            printf ("%d ", i);
    }
    return 0;
}
```

Code Solution in C++

```
#include<iostream>
using namespace std;
int reverse (int);
int main ()
{
    int i, f, l;
    cout << "enter the starting \n", f;
    cin >> f;
    cout << "enter the ending\n", l;
    cin >> l;
    for (i = f; i <= l; i++)
    {
        if (i == reverse (i))
            cout << i << " ";
    }
    return 0;
}
int reverse (int a)
{
    int n = 0, d = 0, rev = 0;
    n = a;
    while (n != 0)
    {
        d = n % 10;
        rev = rev * 10 + d;
        n = n / 10;
    }
    return rev;
}
```

Code Solution in JAVA

```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.Collections;
public class Main
{
    static int palindrome (int no1)
    {
        int rem = 0;
        int div = no1;
        while (div != 0)
        {
            int r = div % 10;
            rem = (rem * 10) + r;
            div = div / 10;
        }
        return rem;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter Upper and Lower Limit");
        int ul = sc.nextInt ();
        int ll = sc.nextInt ();

        for (int i = ul; i <= ll; i++)
        {
            if (i == palindrome (i))
                System.out.print(i+" ");
        }
    }
}
```

Code Solution in PYTHON

```
# Palindrome Number Checking
first_number = int(input())
second_number = int(input())
for i in range(first_number, second_number+1):
    reverse = 0
    temp = i
    while temp != 0:
        remainder = temp % 10
        reverse = (reverse * 10)+remainder
        temp = temp // 10
    if i == reverse:
        print(reverse, end=" ")
```





23. PROGRAM 23

Instructions: You are required to write the code. You can click on compile & run anytime to check the compilation/ execution status of the program. The submitted code should be logically/syntactically correct and pass all the test cases.

Question: The program is supposed to calculate the sum of distance between three points from each other.

For

$x_1 = 1$ $y_1 = 1$

$x_2 = 2$ $y_2 = 4$

$x_3 = 3$ $y_3 = 6$

Distance is calculated as : $\sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$





Code Solution in C

```
#include<stdio.h>
#include<int main()
{
    float x1,y1,x2,y2,x3,y3;
    printf("Enter x1,y1 : ");
    scanf("%f %f",&x1,&y1);
    printf("Enter x2,y2 : ");
    scanf("%f %f",&x2,&y2);
    printf("Enter x3,y3 : ");
    scanf("%f %f",&x3,&y3);

    float firstDiff =(float) sqrt (pow (x2 - x1, 2) + pow (y2 - y1, 2));
    float secondDiff =(float) sqrt (pow (x3 - x2, 2) + pow (y3 - y2, 2));
    float thirdDiff =(float) sqrt (pow (x3 - x1, 2) + pow (y3 - y1, 2));
    printf("%f", (firstDiff + secondDiff + thirdDiff));

    return 0;
}
```

Code Solution in C++

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    float x1, y1, x2, y2, x3, y3;
    cout << "Enter x1,y1: ";
    cin >> x1 >> y1;
    cout << "Enter x2,y2: ";
    cin >> x2 >> y2;
    cout << "Enter x3,y3: ";
    cin >> x3 >> y3;

    float firstDiff = sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2));
    float secondDiff = sqrt(pow(x3 - x2, 2) + pow(y3 - y2, 2));
    float thirdDiff = sqrt(pow(x3 - x1, 2) + pow(y3 - y1, 2));

    cout << firstDiff + secondDiff + thirdDiff;

    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        float x1 = sc.nextFloat();
        float y1 = sc.nextFloat();
        float x2 = sc.nextFloat();
        float y2 = sc.nextFloat();
        float x3 = sc.nextFloat();
        float y3 = sc.nextFloat();
        float firstDiff =(float) Math.sqrt (Math.pow (x2 - x1, 2) + Math.pow (y2 - y1, 2));
        float secondDiff =(float) Math.sqrt (Math.pow (x3 - x2, 2) + Math.pow (y3 - y2, 2));
        float thirdDiff =(float) Math.sqrt (Math.pow (x3 - x1, 2) + Math.pow (y3 - y1, 2));
        System.out.println (firstDiff + secondDiff + thirdDiff);
    }
}
```

Code Solution in PYTHON

```
import math
x1, y1 = 1, 1
x2, y2 = 2, 4
x3, y3 = 3, 6
first_diff = math.sqrt(math.pow(x2-x1, 2) + math.pow(y2-y1, 2))
second_diff = math.sqrt(math.pow(x3-x2, 2) + math.pow(y3-y2, 2))
third_diff = math.sqrt(math.pow(x3-x1, 2) + math.pow(y3-y1, 2))
print(round(first_diff,2), round(second_diff,2), round(third_diff,2))
```





24. PROGRAM 24 - FIND THE MAXIMUM VALUE AND ITS INDEX IN THE ARRAY

Problem Statement:

You are given a function, void MaxInArray(int arr[], int length); The function accepts an integer array ‘arr’ of size ‘length’ as its argument. Implement the function to find the maximum element of the array and print the maximum element and its index to the standard output

(STDOUT). The maximum element and its index should be printed in separate lines.

Note:

- Array index starts with 0
- Maximum element and its index should be separated by a line in the output
- Assume there is only 1 maximum element in the array
- Print exactly what is asked, do not print any additional greeting messages

Example:

Input:

23 45 82 27 66 12 78 13 71 86

Output:

86

9

Explanation:

86 is the maximum element of the array at index 9.





Code Solution in C

```
#include <stdio.h>
#include <limits.h>

void MaxInArray (int arr[], int length)
{
    int max = INT_MIN, index = -1;
    for (int i = 0; i < length; i++)
    {
        if (arr[i] > max)
        {
            max = arr[i];
            index = i;
        }
    }
    printf ("%d\n%d\n", max, index);
}

int main ()
{
    int n;
    scanf ("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++)
        scanf ("%d", &arr[i]);
    MaxInArray (arr, n);
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
void MaxInArray (int arr[], int length)
{
    int max = INT_MIN, index = -1;
    for (int i = 0; i < length; i++)
    {
        if (arr[i] > max)
        {
            max = arr[i];
            index = i;
        }
    }
    cout << max << endl << index;
}
int main ()
{
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n; i++)
        cin >> arr[i];
    MaxInArray (arr, n);
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Main
{
    public static void maxInArray (int arr[], int length)
    {
        int max = arr[0], index = 0;
        for (int i = 1; i < length; i++)
            if (arr[i] > max)
            {
                max = arr[i];
                index = i;
            }
        System.out.println (max);
        System.out.println (index);
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int n = sc.nextInt ();
        int arr[] = new int[n];
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt ();
        maxInArray (arr, arr.length);
    }
}
```





25. PROGRAM 25 - AUTOBIOGRAPHICAL NUMBER

Problem Statement:

An Autobiographical Number is a number N such that the first digit of N represents the count of how many zeroes are there in N, the second digit represents the count of how many ones are there in N and so on.

You are given a function, **def FindAutoCount(n):**

The function accepts string “n” which is a number and checks whether the number is an autobiographical number or not. If it is, an integer is returned, i.e. the count of distinct numbers in ‘n’. If not, it returns 0.

Assumption:

- The input string will not be longer than 10 characters.
- Input string will consist of numeric characters.

Note:

If string is None return 0.

Example:

Input:

n: “1210”

Output:

3

Explanation:

0th position in the input contains the number of 0 present in input, i.e. 1, in 1st position the count of number of 1s in input i.e. 2, in 2nd position the count of 2s in input i.e. 1, and in 3rd position the count of 3s i.e. 0, so the number is an autobiographical number.

Now unique numbers in the input are 0, 1, 2, so the count of unique numbers is 3. So 3 is returned.





Code Solution in C

```
#include <stdio.h>
#include <string.h>

int FindAutoCount (char n[])
{
    int sum = 0;
    int length = strlen (n);
    int uniqueDigits[10] = { 0 };

    for (int i = 0; i < length; i++)
    {
        sum += (n[i] - '0');
        uniqueDigits[n[i] - '0'] = 1;
    }

    if (sum != length)
        return 0;

    int count = 0;
    for (int i = 0; i < 10; i++)
    {
        if (uniqueDigits[i] == 1)
            count++;
    }

    return count;
}

int main ()
{
    char n[100];
    scanf ("%s", n);
    printf ("%d", FindAutoCount (n));
    return 0;
}
```

Code Solution in C++

```
#include<bits/stdc++.h>
using namespace std;
int FinndAutoCount (string n)
{
    int sum = 0;
    set < char >st;
    for (int i = 0; i < n.size (); i++)
    {
        sum += (n[i] - '0');
        st.insert (n[i]);
    }
    if (sum != n.size ())
        return 0;
    return st.size ();
}
int main ()
{
    string n;
    cin >> n;
    cout << FinndAutoCount (n);
    return 0;
}
```

Code Solution in JAVA

```
import java.util.*;
class Solution
{
    public static int findAutoCount (String n)
    {
        int sum = 0;
        for (int i = 0; i < n.length (); i++)
            sum = sum + Integer.parseInt (n.charAt (i) + "");
        if (sum == n.length ())
        {
            int count = 0;
            int arr[] = new int[10];
            for (int i = 0; i < n.length (); i++)
                arr[Integer.parseInt (n.charAt (i) + "")]++;
            for (int i = 0; i < arr.length; i++)
                if (arr[i] != 0)
                    count++;
            return count;
        }
        else
            return 0;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        String str = sc.next ();
        System.out.println (findAutoCount (str));
    }
}
```





26. PROGRAM 26 - BALANCE FRUITS

Implement the following function:

```
int BalanceFruits(int a, int m, int rs);
```

You have a basket full of apples and mangoes, your job is to make the number of apples and given a function that accepts three integers 'a', 'm' and 'rs' as its argument where 'a' and a basket respectively and 'rs' is the rupees that you have. Implement the function to balance the basket.

- If ' $a' > m'$, then buy $(a - m)$ mangoes at the rate of Rs 1 per mango.
- If ' $a' < m'$, then sell $(m - a)$ mangoes at the rate of Rs 1 per mango.

Return the total rupees left with you after balancing the fruits.

Assumption:

- $a \geq 0$, $m \geq 0$ and $rs \geq 0$
- $rs \geq (a - m)$

Note: If $a = m$, return rs unchanged

Example

Sample Input

a : 8

m : 4

Rs : 6

Sample Output

2

Explanation: Since $a > m$, $(a - m)$ mangoes are bought at Rs 1 per mango, so $rs = 6 - 4 = 2$. Thus, output is 2.

Case 1

Input

8 4 6

Output

2

Case 2

Input

7 4 6

Output

3





Code Solution:

```
#include<iostream>
using namespace std;
int main()
{
    int a, m, rs;
    cin >> a >> m >> rs;
    if(a > m)
    {
        cout << rs - (a - m);
    }
    else if( a == m)
    {
        cout << rs;
    }
    else if( a < m)
    {
        cout << rs + (m - a);
    }
}
```





27. PROGRAM 27 - TALLEST TREE

A TreeHeight is represented by the following structure:

```
struct TreeHeight
{
    int feets;
    int inches;
};
```

You are given a function,

```
int TallestTree(struct TreeHeight trees[], int n);
```

The function accepts an array 'trees' of type 'TreeHeight' consisting of 'n' elements as its argument. 'TreeHeight' consists of two integers 'feets' and 'inches' which represents the height of a tree. Implement the function to find the tallest tree among the 'trees' and return its height (in inches).

Height of a tree = $(12 * \text{feets}) + \text{inches}$

Assumption:

- $\text{feets} > 0$
- $0 \geq \text{inches} \geq 11$

Note:

- Computed value lies within integer range.
- Return -1 if trees is null (None, in case of Python).
- $\text{trees}(0)$ is the first element.

Example:

Sample Input:

Feets	Inches
10	4
23	5
21	2
27	7

Sample Output:

33	1
----	---





Explanation:

Height of trees (in inches):

- $\text{trees}[0] = (10 * 12) + 4 = 124$
- $\text{trees}[1] = (23 * 12) + 5 = 281$
- $\text{trees}[2] = (21 * 12) + 2 = 254$
- $\text{trees}[3] = (27 * 12) + 7 = 331$

Maximum of $\{124, 281, 254, 331\} = 331$, thus output is 331.

Case 1**Input**

4

10 4

23 5

21 2

27 7

Output

331

Case 2

2

46 10

55 6

Output

666





Code Solution:

```
#include<iostream>
using namespace std;
struct TreeHeight
{
    int feets;
    int inches;
};
int main()
{
    int n;
    cin >> n;
    struct TreeHeight arr[n];
    for(int i = 0; i < n; i++)
    {
        cin >> arr[i].feets;
        cin >> arr[i].inches;
    }
    int max = 0;
    for(int i = 0; i < n; i++)
    {
        int temp = arr[i].feets * 12 + arr[i].inches;
        if(max < temp)
            max = temp;
    }
    cout << max;
}
```





28. PROGRAM 28 - FRAGMENTS DIVISIBLE BY ELEVEN

You are given a function,

```
def DivisibilityByEleven(num):
```

The function accepts an integer 'num' as input. You have to implement the function such that it returns the number of contiguous integer fragments of 'num' that are divisible by 11. Contiguous integer fragments of a number, say 1273, are:

Example:

Input:

1215598

Output:

4

Explanation: 55, 121, 12155 and 15598 are contiguous fragments of the number 1215598 that are divisible by 11.

Case 1:

Input

1215598

Output

5

Case 2:

Input

55

Output

1





Code Solution:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    cin >> n;
    int temp = n;
    int sum, count = 0;
    vector<int> v;
    int len = 0;
    while(n != 0)
    {
        v.push_back(n % 10);
        n = n / 10;
    }
    reverse(v.begin(), v.end());
    for(int i = 0; i < v.size(); i++)
    {
        sum = v[i];
        for(int j = i+1; j < v.size(); j++)
        {
            sum = sum * 10 + v[j];
        }
        if(sum % 11 == 0)
            count++;
    }
    cout << count; }
```





29. PROGRAM 29 - CONVERT TO PALINDROME

A palindrome is a sequence of character that has the property of reading the same way either in direction. You are given a function,
char *ConvertPalindrome(char* str)

The function accepts the string str. Implement the function to find and return the minimum characters required to append at the end of string str to make it palindrome

Assumption

- 1.String will contain only lower case English alphabets.
- 2.Length of string is greater than or equal to 1

Note:

1. If string is already palindrome return “NULL”.
2. You have to find the minimum characters required to append at the end of string to make it palindrome.

Example:

Sample Input

abcdc

Sample Output

ba

Explanation: If we append ‘ba’ at the end of the string ‘abcdc’ it becomes abcdcba i.e. a palindrome.

Case 1:

Input

abcdc

Output

ba

Case 2:

Input

ab

Output

a





Code Solution:

```
#include<bits/stdc++.h>
using namespace std;
bool checkPalindrome(string s,int i,int j)
{
    while(i<j)
    {
        if(s[i] != s[j])
            return false;
        i++;
        j--;
    }
    return true;
}
int main()
{
    string inp;
    cin>>inp;
    int n=inp.length();
    int i;
    for(i=0;i<n;i++)
    {
        if(checkPalindrome(inp,i,n-1))
            break;
    }
    for(int j = i-1; j >= 0; j--)
        cout << inp[j];
    return 0;
}
```





30. PROGRAM 30 - COUNT SPECIFIC NUMBERS

You are required to implement the following function:

```
int CountSpecificNumbers(int m, int n);
```

The function accept two arguments m and n which are integers. You are required to calculate the count of numbers having only 1, 4 and 9 as their digits between the numbers lying in the range m and n both inclusive, and return the same. Return -1 if m>n.

Example:

Sample Input:

100

200

Sample Output:

9

Explanation: The numbers between 100 and 200, both inclusive having only 1, 4 and 9 as their digits are 111, 114, 119, 141, 144, 149, 191, 194, 199. The count is 9 hence 9 is returned.

Case 1:

Input

100

200

Output

9

Case 2:

Input

1

100

Output

12





Code Solution:

```
#include<bits/stdc++.h>
using namespace std;
int CountSpecificNumber(int m,int n);
int main()
{
    int m,n;
    cin>>m>>n;
    cout<<CountSpecificNumber(m,n);
    return 0;
}
int CountSpecificNumber(int m,int n)
{
    if(m<=n)
    {
        int i,count=0;
        for(i=m;i<=n;i++)
        {
            int num=i,flag=1;
            while(num)
            {
                int n = num%10;
                num=num/10;
                if(n == 1 || n == 4 || n == 9)
                    continue;
                else
                {
                    flag=0;
                    break;
                }
            }
            if(flag==1)
                count++;
        }
    }
    return count;
}
return -1;
}
```





31. PROGRAM 31 - PICKING TICKETS

Consider an array of n tickets prices, tickets . A number, m , is defined as the size of some subsequence, s , of tickets where each element covers an unbroken range of integers. That is to say if you were to sort the elements in s , the absolute difference between any elements j and $j+1$ would be either 0 or 1. Determine the maximum length of a subsequence chosen from the ticket array.

Example:

Tickets = [8,5,4,8,4]

Valid subsequences, sorted, are {4,4,5} and {8,8}. These subsequences have m , values of 3 and 2, respectively. Return 3.

Return:

Int: an integer that denotes the maximum possible value of m .

Constraints:

$1 \leq n \leq 105$

$1 \leq \text{tickets}[i] \leq 109$

Example:

Sample Input:

STDIN	FUNCTION
-----	-----

4	-> tickets [] size n=4
---	------------------------

4	-> tickets = [4,13,2,3]
---	-------------------------

13	
----	--

2	
---	--

3	
---	--

Sample Output:

3	
---	--

Explanation: There are two subsequences of tickets that contain consecutive integers when sorted: {2,3,4} and {13}. These subsequences have m values of 3 and 1, respectively. Return the maximum values of m , which is 3.

Case 1:

Input

4

4

13

2





3

Output

3

Case 2:

Input

5

8 5 4 8 4

Output

3





Code Solution:

```
#include<bits/stdc++.h>
using namespace std;
int maxSubsequence(int a[],int n)
{
    int max=0,i,count=0,flag=0;
    for(i=0;i<n-1;i++)
    {
        int dif = a[i+1]-a[i];
        if(dif==1||dif==0)
        {
            count++;
            flag=1;
        }
        else
        {
            if(count>max)
                max=count;
            count=0;
        }
    }
    if(flag)
        return count>max?count+1:max+1;
    else
        return count;
}
int main()
{
    int n;
    cin>>n;
    int a[n];
    for(int i=0;i<n;i++)
        cin>>a[i];
    sort(a,a+n);
    cout<<maxSubsequence(a,n);
}
```





32. PROGRAM 32 - COUNT OCCURRENCES OF DIGIT

You are required to implement the following function:

```
int CountDigiOccurrences(int l, int u, int x);
```

The function accepts 3 positive integers 'l', 'u' and 'x' as its arguments. You are required to calculate the number of occurrences of a digit 'x' in the digits of numbers lying in the range between 'l' and 'u' both inclusive, and return the same.

Note:

- $l < u$
- $0 < x < 9$

Example:

Sample Input:

I: 2

u: 13

x: 3

Sample Output:

2

Explanation: The number of occurrences of digit 3 in the digits of numbers lying in the range [2, 13] both inclusive is 2, ie (3, 13), hence 2 is returned.

Case 1:

Input

2 13 3

Output

2

Case 2:

Input

1 30 2

Output

13





Code Solution:

```
#include<iostream>
using namespace std;
int countDigitOccurrences(int l, int u, int x)
{
    int rem, count = 0;
    for(int i = l; i <= u; i++)
    {
        int temp = i;
        while(temp != 0 )
        {
            if(temp % 10 == x)
                count++;
            temp = temp / 10;
        }
    }
    return count;
}
int main()
{
    int l, u, x;
    cin >> l >> u >> x;
    cout << countDigitOccurrences(l, u, x);
    return 0;
}
```





33. PROGRAM 33 - FIND THE BMI

A person's body mass index is a simple calculation based on height and weight that classifies the person as underweight, overweight, or normal. The formula for metric unit is,
 $BMI = \text{weight in kilograms} / (\text{height in meters})^2$

You are given a function,

```
Int GetBMICategory(int weight, float height);
```

The function accepts the weight (an integer number) and height (a floating point number) of a person as its arguments. Implement the function such that it calculates the BMI of the person and returns an integer, the person's BMI category as per the given rules:

1. If $BMI < 18$, return 0.
2. If $18 \geq BMI < 25$, return 1.
3. If $25 \geq BMI < 30$, return 2.
4. If $30 \geq BMI < 40$, return 3.
5. If $BMI \geq 40$, return 4.

Note:

Weight > 0 and its unit is kilogram.

Height > 0 and its unit is metre.

Compute BMI as a floating-point.

Example:

Input:

42

1.54

Output:

0

Explanation: The person's weight is 42Kg and height is 1.54 metres $BMI = \text{weight in kilograms} / (\text{height in meters})^2 = 42 / (1.54 * 1.54) = 17.70$

Since, $BMI < 18$, as per given rules, the output is 0.

Sample Input:

62

1.52

Sample Output:

2





Case 1:

Input

42

1.54

Output

0

Case 2:

Input

62

1.52

Output

2





Code Solution:

```
#include <iostream>
using namespace std;
int findBMI(int weight,float height){
    float ans = (float) (weight / (height * height));
    if(ans < 18)
        return 0;
    else if(ans >= 18 && ans < 25)
        return 1;
    else if(ans >= 25 && ans < 30)
        return 2;
    else if(ans >= 30 && ans < 40)
        return 3;
    else
        return 4;
}

int main() {
    int weight;
    float height;
    cin>>weight>>height;
    cout<<findBMI(weight,height);

    return 0;
}
```





34. PROGRAM 34 - NUMEROLOGICAL REDUCTION

Numerological reduction of a number is the sum of digits calculated recursively until the resulting value is a single digit.

You are given a function, int FindNumerologicalReduction(int n);

The function accepts a positive integer 'n'. Implement the function to find and return the numerological reduction of 'n'.

Assumption: $n >= 0$

Example:

Sample Input:

3245

Sample Output:

5

Explanation:

$3245 = 3+2+4+5 = 14$

$14 = 1+4 = 5$

Hence 5 is the numerological reduction of 3245.

Case 1:

Input

3245

Output

5

Case 2:

Input

12345

Output

6





Code Solution:

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cin >> n;
    if(n % 9 == 0)
        cout << "9";
    else
        cout << n % 9;

    return 0;
}
```





35. PROGRAM 35 - EMAIL ID OBFUSCATION

Personal information like email id and other contact details need to be handled in a way so that privacy of the user is safeguarded. One way to do this is to hide or partially hide such information, also called obfuscation of the information.

Given the email id as input, the program should obfuscate the id as follows:

- 1) For the given mail id, the part that comes before @ is referred to as the first part. If there is no first @ character, the mail id is invalid.
- 2) If the first part of the email id is less than or equal to 5 characters in length, replace all characters in the first part with *
- 3) If the first part of the email id is greater than 5 characters in length, print the first 3 characters as it is and replace the remaining characters of the first part with *
- 4) If the email id is invalid, print Invalid Input.

Example 1:

Input :

abc@gmail.com

Output :

***@gmail.com

Explanation: Here the first part is 'abc'. This is less than 5 characters in length, so replace all characters in the first part with *.

Example 2:

Input :

abcdefghijklm@gmail.com

Output :

abc*****@gmail.com

Explanation: Here the first part is 'abcdefghijklm'. This is more than 5 characters in length, so replace all characters in the first part except the first with *.

Complete the function **obfuscateMailId** in the editor. The function must print the obfuscate mail id as per the rules mentioned. If the mail id is invalid, it should print Invalid Input.

obfuscateMailId has the following parameter(s) :

mailid : a string representing the mail id

Constraints :

- 1) A valid mail id should contain the character @ (e.g. myid_at_gmail is invalid)
- 2) A valid mail id should have first part with at least one or more characters (e.g. @myid is invalid)





36. PROGRAM 36 - SUM OF NUMBERS DIVISIBLE BY BOTH 3 AND 5

You are required to implement the following function:

Int Calculate (int m, int n);

The function accepts 2 positive integer ‘m’ and ‘n’ as its arguments. You are required to calculate the sum of numbers divisible both by 3 and 5, between ‘m’ and ‘n’ both inclusive and return the same.

Note: $0 < m \leq n$

Example

Input:

m : 12

n : 50

Output

90

Explanation: The numbers divisible by both 3 and 5, between 12 and 50 both inclusive are {15, 30, 45} and their sum is 90.

Sample Input

m : 100

n : 160

Sample Output

405

Case 1

Input (stdin)

12

50

Output (stdout)

90

Case 2

Input (stdin)

20

30

Output (stdout)

30





Code Solution:

```
#include<iostream>
using namespace std;

int Calculate(int, int);
int main()
{
    int m, n, result;
    cin>>m;
    cin>>n;
    result = Calculate(n,m);
    cout<<result;
    return 0;
}

int Calculate(int n, int m)
{
    int i, sum = 0;
    for(i=m;i<=n;i++)
    {
        if((i%3==0)&&(i%5==0))
        {
            sum = sum + i;
        }
    }
    return sum;
}
```





37. PROGRAM 37 - TABLE OF NUMBERS

Write a program to display the table of a number and print the sum of all the multiples in it.

Case 1

Input (stdin)

12

Output (stdout)

12 24 36 48 60 72 84 96 108 120

660

Case 2

Input (stdin)

5

Output (stdout)

5 10 15 20 25 30 35 40 45 50

275





Code Solution:

```
#include <iostream>
using namespace std;

int main()
{
    int n, i, value=0, sum=0;
    cin>>n;
    for(i=1; i<=10; ++i)
    {
        value = n * i;
        cout<<value<<" ";
        sum=sum+value;
    }
    cout<<endl<<sum ;
return 0;
}
```





38. PROGRAM 38

Robert is expert in strings where he challenges everyone to write a program for the below implementation.

Two strings and comprising of lower case English letters are compatible if they are equal or can be made equal by following this step any number of times:

Select a prefix from the string (possibly empty), and increase the alphabetical value of all the characters in the prefix by the same valid amount.

For example if the string is abc and we select the prefix ab then we can convert it to bcc by increasing the alphabetical value by 1. But if we select the prefix abc then we cannot increase the alphabetical value.

Your task is to determine if given strings are compatible.

Input format

First line: String A

Next line: String B

Output format:

For each test case, print YES if string can be converted to string , otherwise print NO.

Constrain: $1 \leq (\text{len of } A, B) \leq 1000005$

Sample Input:

abaca

cdbda

Sample Output:

YES

Explanation: The string abaca can be converted to bcbda in one move and to cdbda in the next move.

Test Case 1

Input

abaca cdbda

Output

YES

Test Case 2

Input

abcd aabcd

Output

NO





Code Solution:

```
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
    char str1[1000005];
    char str2[1000005];
    cin>>str1>>str2;
    int max = str2[0] - str1[0];
    if(max > 0)
    {
        for(int i = 1; i < strlen(str1); i++)
        {
            int x = str2[i] - str1[i];
            if(x > max)
            {
                cout<<"NO";
                return 0;
            }
        }
        cout<<"YES";
    }
    else
        cout<<"NO";
    return 0;
}
```





39. PROGRAM 39 - SECOND GREATEST ELEMENT

Write a program to print the second greatest element in the given array. In case, if the second largest element is not present in the array, print "There is no second largest element" and if the size of the array is less than 2, print "Invalid Input".

Sample Input:

5
12 5 7 3 90

Sample Output:

12

Case 1

Input (stdin)

5
12 5 7 3 90

Output (stdout)

12

Case 2

Input (stdin)

10
10 20 50 43 60 51 70 82 93 55

Output (stdout)

82





Code Solution:

```
#include<iostream>
using namespace std;
#include <limits.h>

void print2largest(int arr[], int arr_size)
{
    int i, first, second;
    if (arr_size < 2)
    {
        cout<<"Invalid Input";
        return;
    }
    first = second = INT_MIN;
    for (i = 0; i < arr_size ; i++)
    {
        if (arr[i] > first)
        {
            second = first;
            first = arr[i];
        }
        else if (arr[i] > second && arr[i] != first)
            second = arr[i];
    }

    if (second == INT_MIN)
        cout<<"There is no second largest element";
    else
        cout<<second;
}

int main()
{
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++)
        cin>>arr[i];
    print2largest(arr, n);
    return 0;
}
```





40. PROGRAM 40 - MOST OCCURRING CHARACTER

Write a program to find the most occurring character in the string.

Sample Input:

Happy coding

Sample Output:

p

Case 1

Input (stdin)

Happy coding

Output (stdout)

p

Case 2

Input (stdin)

programming is good

Output (stdout)

p





Code Solution:

```
#include<iostream>
using namespace std;
#include <string.h>

int main()
{
    char str[100], result;
    int i, len;
    int max = -1;
    int freq[256] = {0};
    scanf("%[^\\n]s",str);
    len = strlen(str);
    for(i = 0; i < len; i++)
    {
        freq[str[i]]++;
    }
    for(i = 0; i < len; i++)
    {
        if(max < freq[str[i]])
        {
            max = freq[str[i]];
            result = str[i];
        }
    }

    cout<<result;
    return 0;
}
```





41. PROGRAM 41 - INTEGER DIFFERENCE

Given an array arr and two integer value n and m as input, take the first element and find the difference between the first element and the integer value n. If the difference is less than m then increment that particular array element. Do this for all the element in an array and print the final modified array as output.

Sample Input:

5
2 1 4 5 7
3 2

Sample Output:

3 2 5 5 7

Case 1

Input (stdin)

5
2 1 4 5 7
3 2

Output (stdout)

3 2 5 5 7

Case 2

Input (stdin)

5
1 2 3 4 5
3 5

Output (stdout)

2 3 4 5 6





Code Solution:

```
#include<iostream>
using namespace std;

int result (int arr[], int x, int n,int m);

int main ()
{
    int x, i, n,m;
    cin>>x;
    int arr[x];
    for (i = 0; i < x; i++)
    {
        cin>>arr[i];
    }
    cin>>n;
    cin>>m;
    result (arr, x, n, m);
    for(i=0;i<x;i++)
    {
        cout<<arr[i]<<" ";
    }
    return 0;
}

int result (int arr[], int x, int n,int m)
{
    int i;
    for (i = 0; i < x; i++)
    {
        if ( (arr[i]-n ) < m)
        {
            arr[i]=arr[i]+1;
        }
    }
}
```





42. PROGRAM 42 - SEARCH AN ELEMENT

Write a program to search an element in an array and print its index value. If the element is not present, then print -1.

Sample Input:

5

4 6 7 2 8

7

Sample Output:

index = 2

Case 1

Input (stdin)

5

4 6 7 2 8

7

Output (stdout)

index = 2

Case 2

Input (stdin)

5

4 6 3 2 8

5

Output (stdout)

-1





Code Solution:

```
#include<iostream>
using namespace std;

int search (int arr[], int ele, int n);

int main ()
{
    int n, i, ele;
    cin>>n;
    int arr[n];
    for (i = 0; i < n; i++)
    {
        cin>>arr[i];
    }
    cin>>ele;
    search (arr, ele, n);
    return 0;
}

int search (int arr[], int ele, int n)
{
    int i, flag = 0;

    for (i = 0; i < n; i++)
    {
        if (ele == arr[i])
        {
            cout<<"index = "<< i;
            flag = 1;
            break;
        }
    }

    if (flag == 0)
    {
        cout<<"-1";
    }
}
```





43. PROGRAM 43 - DECIMAL TO BINARY EQUIVALENT

Write a program to convert a decimal number to a binary equivalent number.

Sample Input:

10

Sample Output:

1010

Case 1

Input (stdin)

10

Output (stdout)

1010

Case 2

Input (stdin)

2

Output (stdout)

10





Code Solution:

```
#include<iostream>
using namespace std;

int main()
{
    int a[50],n,i=0;
    cin>>n;
    while(n>0)
    {
        a[i++]=n%2;
        n=n/2;
    }
    for(i=i-1;i>=0;i--)
        cout<<a[i];
    return 0;
}
```





44. PROGRAM 44 - BINARY TO DECIMAL EQUIVALENT

Write a program to convert a binary number to an equivalent decimal value.

Sample Input:

1010

Sample Output:

10

Case 1

Input (stdin)

1010

Output (stdout)

10

Case 2

Input (stdin)

11111

Output (stdout)

31





Code Solution:

```
#include<iostream>
using namespace std;

int main()
{
    int num, binary, decimal = 0, base = 1, rem;
    cin>>num;
    binary = num;
    while (num > 0)
    {
        rem = num % 10;
        decimal = decimal + rem * base;
        num = num / 10 ;
        base = base * 2;
    }
    cout<<decimal;
    return 0;
}
```





45. PROGRAM 45 - INSERT AN ELEMENT

Write a program to create a dynamic array and insert an element in it, in the specified position.

Input Format

First line consists of integer value N

Second line consists of an array elements

Third line consists of integer value as position and value

Output Format

Output consists of an array element

Refer the sample output for formatting

Sample Input:

5

1 2 3 4 5

2 10

Sample Output:

1 10 2 3 4 5

Case 1

Input (stdin)

5

1 2 3 4 5

2 10

Output (stdout)

1 10 2 3 4 5

Case 2

Input (stdin)

5

1 7 8 4 5

2 20

Output (stdout)

1 20 7 8 4 5





Code Solution:

```
#include <iostream>
#include<cstdlib>
using namespace std;

int main()
{
    int *array, position, i, n, value;
    cin>>n;
    array = (int*) malloc(n * sizeof(int));
    if(array == NULL)
    {
        cout<<"Error! memory not allocated.";
        exit(0);
    }
    for(i = 0; i < n; ++i)
    {
        cin>>array[i];
    }
    cin>>position>>value;

    for (i = n - 1; i >= position - 1; i--)
        array[i+1] = array[i];

    array[position-1] = value;

    for (i = 0; i <= n; i++)
        cout<<array[i]<<" ";
    return 0;
}
```





46. PROGRAM 46 - DELETE AT SPECIFIC POSITION

Write a program to create a dynamic array and delete an element from an array from the specified position. If deletion is not possible, print "Deletion not possible."

Sample Input:

5

1 2 3 4 5

2

Sample Output:

1 3 4 5

Case 1

Input (stdin)

5

1 2 3 4 5

2

Output (stdout)

1 3 4 5

Case 2

Input (stdin)

5

1 2 3 4 5

5

Output (stdout)

1 2 3 4





Code Solution:

```
#include <stdio.h>
#include<stdlib.h>
int main()
{
    int *array, position, i,c, n, value;
    scanf("%d", &n);
    array = (int*) malloc(n * sizeof(int));
    if(array == NULL)
    {
        printf("Error! memory not allocated.");
        exit(0);
    }
    for(i = 0; i < n; ++i)
    {
        scanf("%d", &array[i]);
    }
    scanf("%d", &position);

    if ( position >= n+1 )
        printf("Deletion not possible.\n");

    else
    {
        for ( c = position - 1 ; c < n - 1 ; c++ )
            array[c] = array[c+1];
        for( c = 0 ; c < n - 1 ; c++ )
            printf("%d ", array[c]);
    }
    return 0;
}
```





47. PROGRAM 47 - SEGREGATE 0's and 1's

Write a program to segregate all the 0's in the left side and 1's in the right side in the same array.

Sample Input:

5

0 1 0 1 0

Sample Output:

0 0 0 1 1

Case 1

Input (stdin)

5

0 1 0 1 0

Output (stdout)

00011

Case 2

Input (stdin)

6

1 1 0 0 0 0

Output (stdout)

000011





Code Solution:

```
#include<iostream>
using namespace std;

void segregate0and1(int arr[],int n)
{
    int left = 0, right = n-1;
    while(1)
    {
        if(left >= right)
            break;
        if(arr[left] == 0)
            left++;
        else if(arr[right] == 1)
            right--;
        else
        {
            arr[left] = 0;
            arr[right] = 1;
            left++;
            right--;
        }
    }
}

int main()
{
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++)
        cin>>arr[i];
    segregate0and1(arr, n);
    for(int i=0;i<n;i++)
        cout<<arr[i];
    return 0;
}
```





48. PROGRAM 48 - INSERTION SORT

Write a program to create a dynamic array and sort the elements in it using Insertion sort.

Input Format

First line consists of integer value N

Second line consists of an array element

Output Format

Output consists of an array element

Refer the sample output for formatting

Sample Input:

6

1 2 3 4 3 3

Sample Output:

1 2 3 3 3 4

Case 1

Input (stdin)

6

1 2 3 4 3 3

Output (stdout)

1 2 3 3 3 4

Case 2

Input (stdin)

7

8 1 2 5 6 4 2

Output (stdout)

1 2 2 4 5 6 8





Code Solution:

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main()
{
    int n, i, j, *ptr, temp;
    cin>>n;
    ptr = (int*) malloc(n * sizeof(int));
    if(ptr == NULL)
    {
        cout<<"Error! memory not allocated.";
        exit(0);
    }
    for(i = 0; i < n; ++i)
    {
        cin>>ptr[i];
    }
    for(i=1;i<n;i++)
    {
        temp = ptr[i];
        j=i-1;
        while(temp < ptr[j] && j>=0)
        {
            ptr[j+1] = ptr[j];
            --j;
        }
        ptr[j+1]=temp;
    }
    for(i=0; i<n; i++)
        cout<<ptr[i]<<" ";
    free(ptr);
    return 0;
}
```





49. PROGRAM 49 - SORTING THE FIRST HALF OF THE ARRAY IN ASCENDING ORDER AND THE SECOND HALF IN DESCENDING ORDER

Write a program to sort the first half of an array in ascending order and the second half in descending order.

Input Format:

The first line contains an integer 'N', denoting the size of the array.

The next line contains space-separated integers denoting the elements of the array.

Output Format:

Print sorted array

Sample Input:

6

1 2 3 4 5 6

Sample Output:

1 2 3 6 5 4

Case 1

Input (stdin)

9

5 2 4 7 9 3 1 6 8

Output (stdout)

2 4 5 7 9 8 6 3 1

Case 2

Input (stdin)

6

10 30 20 40 60 50

Output (stdout)

10 20 30 60 50 40





Code Solution:

```
#include<iostream>
using namespace std;
int insertion_sort(int n,int list[]);
int main()
{
    int n;
    cin>>n;
    int list[n];
    for(int index = 0;index < n; index++)
    {
        cin>>list[index];
    }
    insertion_sort(n,list);
    return 0;
}
int insertion_sort(int n,int list[])
{
    for (int idx1 = 1; idx1 < n/2; idx1++)
    {
        int key = list[idx1];
        int idx2 = idx1 - 1;
        while((idx2 >= 0) && (list[idx2] > key))
        {
            list[idx2 + 1] = list[idx2];
            idx2--;
        }
        list[idx2 + 1] = key;
    }
    for (int idx1 = n/2; idx1 < n; idx1++)
    {
        int key = list[idx1];
        int idx2 = idx1 - 1;
        while((idx2 >= n/2) && (list[idx2] < key))
        {
            list[idx2 + 1] = list[idx2];
            idx2--;
        }
        list[idx2 + 1] = key;
    }
    for(int i = 0; i < n; i++)
    {
        cout<< list[i]<<" ";
    }
    return 0;
}
```





50. PROGRAM 50 - REGULAR EXPRESSION

Write a program to check the regular expression matching. Hint : '+' indicates consecutive multiple occurrences of that particular character.

Input Format

First line consists of a string s

Second line consists of string s

Output Format

Output consists of string whether it is matched or not

Refer the sample output for formatting

Sample Input:

a+b+c

aaabbc

Sample Output:

Matched

Case 1

Input (stdin)

a+b+c

aaabbc

Output (stdout)

Matched

Case 2

Input (stdin)

b+c+x

bbcxxz

Output (stdout)

Not Matched





Code Solution:

```
#include<iostream>
using namespace std;
int main()
{
    char arr[20],ans[20];
    int i=0,j=0,len;
    cin>>arr>>ans;
    for(len=0;arr[len]!='\0';len++);
    while(arr[i]==ans[j] && arr[i]!='\0')
    {
        if(arr[i+1]=='+' )
        {
            for( ; ans[j]==arr[i] ; j++)
            i=i+2;
        }
        else
        {
            i++;
            j++;
        }
    }
    if(i>=len && ans[j]=='\0')
    {
        cout<<"Matched";
    }
    else
    {
        cout<<"Not Matched";
    }
    return 0;
}
```



51. PROGRAM 51

In this lockdown a family of N members decided to play a game the rules of which are :-

- All N members are made to sit uniformly in a circle (ie. from 1 to N in clockwise direction).
- The game start with the person sitting at first position.
- A song is played in the background. The lyrics of the song are denoted by a string which consists of only letters 'x' and 'y'. Assume that each lyric of the song is a single letter.
- If the lyric 'x' occurs in the song, the member who is currently holding the Parcel passes it on to the next member. This passing takes place in clockwise direction.
- If the lyric 'y' occurs in the song, the member who is currently holding the Parcel loses his/her chances of winning the game. He/she hands over the parcel to the next member (in clockwise direction) and moves out.
- The game continues until a single member survives in the end. He/she will be the winner of the game.
- Note that the song repeats continuously ie. while the game is going on, if at all the song ends, the stereo system will automatically start playing the song from the start without any delay.

You have to find out the member who wins the game.

Input: The input consists of 2 lines. The first line consists of N , the member of family in the class. The next line consists of a string denoting the lyrics of the song the teacher plays.

Output: Print a single integer denoting the roll number of the student who wins the game.

Constraints:

$2 \leq N \leq 100000$

$1 \leq |S| \leq 100000$, where $|S|$ denotes the length of the input string. It is guaranteed that at least 1 lyric in the song will be a 'y'

Sample Input:

3

xyx

Sample Output:

1

Explanation:

Starting from 1 lyrics : 'x' therefore he passes the ball to 2nd

2nd turn lyrics : 'y' therefore 2nd member gets out of game and passes to 3rd

3rd turn lyrics : 'x' therefore 3rd passes ball to first.

4th turn lyrics : 'x' passes to 3rd





5th turn lyrics: 'y' therefore gets eliminated.

Hence the person sitting at position 1 won this game.

Test Case 1

Input

3 xyx

Output

1

Test Case 2

Input

6 xxyyxy

Output

2





Code Solution:

```
#include<bits/stdc++.h>
#include<string>
using namespace std;
#define ll long long
ll n,slen;

void fun (ll size,ll k,string s, ll l, ll *ingame)
{
    if(size==1)
    {
        ll i=1;
        while(ingame[i]!=1)
        {i++;

        }
        cout<<i<<endl;
        return;
    }
    if(l==slen) l=0;
    if(k==n+1) k=1;
    if(ingame[k]==0)
    {

        fun(size,k+1,s,l,ingame);
    }
    else
    {
        if(s[l]=='x')
        fun(size,k+1,s,l+1,ingame);
        else{
            ingame[k]=0;
            fun(size-1,k+1,s,l+1,ingame);
        }
    }
    return;
}

int main()
{
```





```
cin>>n;
string s;
cin>>s;
slen=s.length();
ll ingame[n+1];
for(int i =1;i<=n;i++)
{
    ingame[i]=1;
}
fun(n,l,s,0,ingame);
}
```





52. PROGRAM 52

Write a program to find the equilibrium element in the given array.

Hint : A[i] is equilibrium element if $A[0] + A[1] + \dots + A[i-1] == A[i+1] + A[i+2] + \dots + A[n-1]$

Sample Input 1:

6

1 2 3 4 3 3

Sample Output 1:

4

Sample Input 2:

6

5 6 4 2 3 1

Sample Output 2:

No Equilibrium element found

Test Case 1

Input

6 1 2 3 4 3 3

Output

4

Test Case 2

Input

6 5 6 4 2 3 1

Output

No Equilibrium element found





Code Solution:

```
#include<iostream>
using namespace std;

int findElement(int arr[], int n)
{
    int prefixSum[n];
    prefixSum[0] = arr[0];
    for (int i = 1; i < n; i++)
        prefixSum[i] = prefixSum[i - 1] + arr[i];
    int suffixSum[n];
    suffixSum[n - 1] = arr[n - 1];
    for (int i = n - 2; i >= 0; i--)
        suffixSum[i] = suffixSum[i + 1] + arr[i];
    for (int i = 1; i < n - 1; i++)
        if (prefixSum[i] == suffixSum[i])
            return arr[i];
    return -1;
}

int main()
{
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++)
        cin>>arr[i];
    int result = findElement(arr, n);
    if(result == -1)
        cout<<"No Equilibrium element found";
    else
        cout<<result;
    return 0;
}
```





53. PROGRAM 53 - NON REPEATING ELEMENTS

Write a program to eliminate the common elements in the given 2 arrays and print only the non repeating elements and the total.

Sample Input:

```
5 4  
1 2 8 6 5  
2 6 8 10
```

Sample Output:

```
1 5 10  
3
```

Case 1

Input (stdin)

```
5 4  
1 2 8 6 5  
2 6 8 10
```

Output (stdout)

```
1 5 10  
3
```

Case 2

Input (stdin)

```
5 5  
10 20 30 40 50  
60 70 80 90 50
```

Output (stdout)

```
10 20 30 40 60 70 80 90  
8
```





54. PROGRAM 54 - FIND THE Nth ELEMENT

Write a program to create a Singly Linked List and find the Nth element from the end of the list.

Hint: Insertion at the beginning.

Output: output consists of single line, print the element in the given position in a list, else print "**No node found**"

Sample Input:

```
5  
10 8 6 4 0  
2
```

Output:

```
8
```

Case 1

Input (stdin)

```
5  
10 8 6 4 0  
2
```

Output (stdout)

```
8
```

Case 2

Input (stdin)

```
10  
100 200 300 400 500 600 700 800 900 101  
11
```

Output (stdout)

```
No node found
```





Code Solution:

```
#include<iostream>
#include<stdlib.h>
using namespace std;
struct Node
{
    int num;
    struct Node* link;
};
void insertAtBeginning(struct Node** head_ref, int new_data)
{
    struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));
    new_node->num = new_data;
    new_node->link = (*head_ref);
    (*head_ref) = new_node;
}
int printNthFromLast(struct Node* head, int n)
{
    int len = 0, i;
    struct Node *temp = head;
    while (temp != NULL)
    {
        temp = temp->link;
        len++;
    }
    if (len < n)
        return 0;
    temp = head;
    for (i = 1; i < len-n+1; i++)
        temp = temp->link;
    cout<<temp->num;
    return 1;
}
int main()
{
    struct Node* head = NULL;
    int total_elem_to_insert, tmp, i,n;
    cin>>total_elem_to_insert;

    for(i=0;i<total_elem_to_insert;i++)
    {
```





```
cin>>tmp;
insertAtBeginning(&head, tmp);
}
scanf("%d",&n);

if(0==(printNthFromLast(head, n)))
cout<<"No node found"<<endl;
return 0;
}
```





55. PROGRAM 55 - ANAGRAM

Write a program to find whether the given string is the anagram of the first string.

Sample Input 1:

eat

ate

Sample Output 1:

Anagram

Sample Input 2:

eat

abc

Sample Output 2:

Not Anagram

Case 1

Input (stdin)

eat

ate

Output (stdout)

Anagram

Case 2

Input (stdin)

eat

abc

Output (stdout)

Not Anagram





Code Solution:

```
#include<iostream>
using namespace std;

int find_anagram(char [], char []);
int main()
{
    char array1[100], array2[100];
    int flag;
    cin>>array1;
    cin>>array2;
    flag = find_anagram(array1, array2);
    if (flag == 1)
        cout<<"Anagram";
    else
        cout<<"Not Anagram";
    return 0;
}
int find_anagram(char array1[], char array2[])
{
    int num1[26] = {0}, num2[26] = {0}, i = 0;
    while (array1[i] != '0')
    {
        num1[array1[i] - 'a']++;
        i++;
    }
    i = 0;
    while (array2[i] != '0')
    {
        num2[array2[i] - 'a']++;
        i++;
    }
    for (i = 0; i < 26; i++)
    {
        if (num1[i] != num2[i])
            return 0;
    }
    return 1;
}
```





56. PROGRAM 56 - MERGE TWO SORTED LINKED LIST

Write a program to merge two sorted linked lists. Given two sorted linked lists say list1 and list2, merge list2 with list1. Insertion has to be done at the end of the list.

Input Format

The first line consists of the size of the first sorted linked list.

The second line consists of list of elements of first sorted linked list separated by space.

The third line consists of the size of the second sorted linked list.

The fourth line consists of list of elements of second sorted linked list separated by space.

Output Format: The output should be the merged list of two linked list.

Refer the sample input & output for formatting specifications.

Sample Input

3

1 3 5

3

2 4 6

Sample Output

1->2->3->4->5->6->NULL

Case 1

Input (stdin)

3

1 3 5

3

2 4 6

Output (stdout)

1->2->3->4->5->6->NULL





57. PROGRAM 57 - APPEND WITHOUT DUPLICATES

John assigned a roll number to each student in ascending order. Every time he assigns a number to a student he wants to check whether the number is already assigned to any other student to avoid the duplicates in the roll number. Add the number to the list only if the number is not present already. Implement this concept using a Linked List. If the list is empty, print "List is empty".

Input Format: The input consists of a list of integers, negative value indicates the end of the linked list.

Output Format: The output should be numbers in the list in a separate line.

Refer the sample input & output for formatting specifications.

Sample Input

```
11  
22  
33  
22  
33  
44  
-77
```

Sample Output

```
11  
22  
33  
44
```

Case 1

Input (stdin)

```
11  
22  
33  
22  
33  
44  
-77
```





Output (stdout)

11

22

33

44

Case 2**Input (stdin)**

-5

Output (stdout)

List is empty





Code Solution:

```
#include<iostream>
using namespace std;
class Node
{
public:
    int data;
    Node *next;
};

void append(Node **headadd,int data)
{
    Node *temp,*newnode,*temp1;
    temp = *headadd;
    temp1 = *headadd;
    int flag = 1;
    while(temp1 != NULL)
    {
        if(temp1->data == data)
            flag = 0;
        temp1 = temp1->next;
    }
    if(flag == 1)
    {
        newnode = new Node();
        newnode->data = data;
        newnode->next = NULL;
        if(*headadd == NULL)
            *headadd = newnode;
        else
        {
            while(temp->next != NULL)
                temp = temp->next;
            temp->next = newnode;
        }
    }
}
void display(Node *head)
{
    if(head == NULL)
    {
```





```
cout<< "List is empty";
return;
}
while(head != NULL)
{
    cout<<head->data<<endl;
    head = head->next;
}
int main()
{
    Node *head = NULL;
    int data;
    do
    {
        cin>>data;
        if(data>0)
            append(&head,data);
    }while(data>0);
    display(head);
    return 0;
}
```

