accenture

specific training

Programming

terv

accenture
specific training

1. Small Large Sum

Write a function SmallLargeSum(array) which accepts the array as an argument or parameter, that performs the addition of the second largest element from the even location with the second largest element from an odd location?

Rules:

All the array elements are unique.
If the length of the array is 3 or less than 3, then return 0.
If Array is empty then return zero.
Sample Test Case 1:

Input:

6

3 2 1 7 5 4

Output:

7

Explanation: The second largest element in the even locations (3, 1, 5) is 3. The second largest element in the odd locations (2, 7, 4) is 4. So the addition of 3 and 4 is 7. So the answer is 7.

Sample Test Case 2:

Input:

7

4 0 7 9 6 4 2

Output:

10

terv

Solution: At first we've to take input from the user i.e in the main function. Then we've to create two different arrays in which the first array will contain all the even position elements and the second one will contain odd position elements. The next step is to sort both the arrays so that we'll get the second-largest elements from both the arrays. At last, we've to add both the second-largest elements that we get from both the arrays. Display the desired output in the main function.

```
#include <bits/stdc++.h>
using namespace std;
int smallLargeSum(int *arr, int n) {
  if(n <= 3) {
    return 0;
  }
  //Here we use vector because we don't know the array size,
  //we can use array also but vector gives us more functionality than
array
  vector<int> arrEven, arrOdd;
  //Break array into two different arrays even and odd
  for(int i = 0; i < n; i++) {
    //If Number is even then add it into even array
    if(i % 2 == 0) {
      arrEven.push_back(arr[i]);
    }
    else {
      arrOdd.push_back(arr[i]);
   }
  }
 //Sort the even array
  sort(arrEven.begin(), arrEven.end());
  //We use sort function from C++ STL library
  //Sort the odd array
  sort(arrOdd.begin(), arrOdd.end());
```

```
 //Taking second largest element from both arrays and add them
   return arrEven[1] + arrOdd[1];
}
// Driver code
int main()
{
   int n;
   cout<<"Enter How many elements you want to enter?\n";
   cin>>n;
   int arr[n];
   //Get input from user
   cout<<"Start entering the numbers\n";
   for(int i = 0; i < n; i++) {
      cin>>arr[i];
   }
   cout<<"Output is\n";
   cout<<smallLargeSum(arr, n);
   return 0;
}
```

2. Check Password

Write a function CheckPassword(str) which will accept the string as an argument or parameter and validates the password. It will return 1 if the conditions are satisfied else it'll return 0?

The password is valid if it satisfies the below conditions:

It should contain at least 4 characters.
At least 1 numeric digit should be present.
1 Capital letter should be there.
Password should not contain space or slash.
The starting character should not be a number.

Sample Test Case:

Input:

bB1_89

Output:

1

```
#include<iostream>
#include<string.h>
using namespace std;
int CheckPassword(char str[]) {
int len = strlen(str);
bool isDigit = false, isCap = false, isSlashSpace=false,isNumStart=false;
//RULE 1: At least 4 characters in it
if (len < 4)
return 0;
for(int i=0; i<len; i++) {
//RULE 2: At least one numeric digit in it
if(str[i]>='0' && str[i]<='9') {
isDigit = true;
}
//RULE 3: At least one Capital letter
else if(str[i]>='A'&&str[i]<='Z'){
isCap=true;
}
//RULE 4: Must not have space or slash
if(str[i]==' '|| str[i]=='/')
isSlashSpace=true;
}
```

```
//RULE 5: Starting character must not be a number
isNumStart = (str[0]>='0' && str[0]<='9');
//FYI: In C++, if int data type function returns the true then it prints 1
and if false then it prints 0
return isDigit && isCap && !isSlashSpace && !isNumStart;
}
int main() {
char password[100];
cout<<"Enter the Password\n";
cin>>password;
cout<<"The output is =\n";
cout<<CheckPassword(password);
}
```

3. Calculate Binary Operations

Write a function CalculateBinaryOperations(str) that accepts the string as an argument or parameter. The string should contains the binary numbers with their operators OR, AND, and XOR?

A Means the AND Operation.
B Means the OR Operation.
C Means the XOR Operation.
By scanning the given string from left to right you've to calculate the string and by taking one operator at a time then return the desired output.

Conditions:

The priority of the operator is not required.
The length of the string is always Odd.
If the length of the string is null then return -1.
Sample Test Case:

Input:

1C0C1C1A0B1

Output:

1

Explanation:

The entered input string is 1 XOR 0 XOR 1 XOR 1 AND 0 OR 1.

Now calculate the string without an operator priority and scan the string characters from left to right. Now calculate the result and return the desired output.

Note: This will convert the char into the num (char – '0') in the c++ language.

```
#include <bits/stdc++.h>
using namespace std;
int CalculateBinaryOperations(char* str)
{
  int len = strlen(str);
  //Let's consider the first element as a answer (because string can be a single char)
  int ans = str[0]-'0';
  for(int i=1; i<len-1; i+=2)
  {
    int j=i+1;
    //Performing operation for AND
    if(str[i]=='A')
    {
     ans = ans & (str[j]-'0');
```

```
 }
    //Performing operation for OR
    else if(str[i]=='B')
    {
      ans = ans | (str[j]-'0');
    }
    //Performing operation for XOR
    else if(str[i]=='C')
    {
      ans = ans ^ (str[j]-'0');
    }
  }
  return ans;
}
int main()
{
  char str[100];
  cout<<"Enter the String:\n";
  cin>>str;
  cout<<"The output is :\n";
  cout<<CalculateBinaryOperations(str);
}
```

4. Find Maximum In An Array

Write a function FindMaxInArray, which will find the greatest number from an array with its desired index? The greatest number and its desired index should be printed in separate lines.

Sample Test Case:

Input:

10

15 78 96 17 20 65 14 36 18 20

Output:

96

2

```
#include<iostream>
using namespace std;
void FindMaxInArray(int arr[],int length)
{
  int max=-1, maxIdx=-1;
  for(int i = 0;i < length; i++)
  {
    if(arr[i] > max)
    {
      max = arr[i];
      maxIdx = i;
    }
  }
  cout<<"The Maximum element in an array is = \n";
  cout<<max;
  cout<<"\nAt the Index = \n";
  cout<<maxIdx;
}
int main()
{
  int n;
```

```
 cout<<"How many elements you want to enter:\n";
   cin>>n;
   int a[n];
   cout<<"Enter elements: \n";
   for(int i=0;i<n;i++)
      cin>>a[i];
   FindMaxInArray(a,n);
}
```

5. Operation Choices

Write a function OperationChoices(c, a, b) which will accept three integers as an argument, and the function will return:

(a + b) if the value of c=1.
(a – b) if the value of c=2.
(a * b) if the value of c=3.
(a / b) if the value of c=4.

Sample Test Case:

Input:

2

15

20

Output:

-5

Here, the value of the c is two i.e 2. So it'll perform the operation as subtraction (15, 20) and will return -5.

```
#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;
}
}
int main()
{
int x, y, z;
int result;
cout<<"Enter c\n";
cin>>x;
cout<<"Enter two elements\n";
cin>>y;
```

```
cin>>z;
result = operationChoices(x, y, z);
cout<<"The result is ";
cout<<result;
}
```

6. Difference Of Sum

Write a function differenceofSum(a,b) which will take two integers as an argument. You've to obtain the total of all the integers ranging from 1 to n (both inclusive) that are not divisible by b. You should also return the distinction between the sum of the integers which are not divisible by b with the sum of the integers divisible by b?

Consider: a and b are greater than 0. i.e a>0 and b>0. And their sum should lies between the integral range.

Sample Test Case 1:

Input:

a = 6 and b = 30

Output:

285

Explanation:

The integers that are divisible by 6 are 6, 12, 18, 24, and 30 and their sum is 90. The integers that are not divisible by 6 are 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23, 25, 26, 27, 28 and 29. And their addition is 375.

The difference between them is (375 – 90) = 285.

Sample Test Case 2:

Input:

a = 10

b = 3

Output:

19

7. Anagram Strings

Write a function to check whether the given strings are anagrams or not. If the given strings are anagram then return ‘yes’ otherwise return ‘no’?

Sample Test Case 1:

Input:

1st: learn

2nd: simple

Output:

no

Sample Test Case 2:

Input:

1st: Listen

2nd: Silent

Output:

yes

Explanation:

The two strings Listen and Silent are anagrams because rearranging all the characters from the second string forms the first string.

8. Product Small Pair

Write a function Productsmallpair(sum, arr) which will accept the two integers sum and arr. These two integers will be used to find the arr(j) and arr(k) where k is not equal to j.arr(j) and arr(k). k != j.arr(j) and arr(k) should be the smallest elements from the array.

Rules:

If the value of n<2 or empty, then return -1.
If these pairs are not found then return the value as 0.
You should make sure that all the values are between the range of integers.

Sample Test Case 1:

Input:

sum: 9

arr: 5 4 2 3 9 1 7

Output:

2

Solution:

From the given array of integers, you've to select the two smallest integers which are 1 and 2. The addition of these two numbers is (1 + 2 = 3) which is less than 9(3 < 9). And the product of these two is 2 (2 x 1 = 2) so, the output we get is 2.

Sample Test Case 2:

Input:

sum: 4

arr: 9 8 -7 3 9 3

Output:

-21

accenture
specific training

9. Replace Character

Write a function Replacecharacter(Char str1, Char ch1, Int 1, Char ch2) which has a string(str) and the two characters ch1 and ch2. Execute a function in such a way that string str will return to its original string, and all the events in ch1 are replaced by ch2 and vice versa?

Consider: The strings will have only alphabets in lower case.

Sample Test Case:

Input:

str: tervpro

ch1: e

ch2: p

Output:

tprvero

Solution:

All the ‘e’s in the string are replaced with the ‘p’ and ‘p’ is replaced with the ‘e’.

terv

10. Reverse a String

Write a function that will accept strings from the user and will reverse the string word-wise. The last word will come as the first word in the output and vice versa?

Sample Test Case 1:

Input:

terv pro

Output:

pro terv

Explanation:

The reverse string word-wise function is applied.

Sample Test Case 2:

Input:

Welcome to Sunday samayal

Output:

samayal Sunday to Welcome

accenture
specific training

11. Question

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 <= i.
Note:
1.Return -1 if the array is null
2.Return 0 if the total amount of food from all houses is not sufficient for all the rats.
3.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 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.

terv

C++ Solution

```
#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;
}
```

JAVA SOLUTION

```
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));
   }
}
```

PYTHON SOLUTION

```
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))
```

12. QUESTION – 12

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.

C++ SOLUTION

```
#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);
}
```

JAVA SOLUTION

```
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));
   }
}
```

PYTHON SOLUTION

```
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))
```

13. DECODE

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 o) 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 upto 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:

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

Assumption:
1 < n < = 36

Example

Input

n: 12
num: 718

Output

4BA

Explanation

num = 718, divisor = 12, quotient=59, remainder=10(A).
num = 59, divisor = 12, quotient=4, remainder=11(B).
num = 4, divisor = 12, quotient=0, remainder=4(A).
Sample Input
n: 21
num: 5678

Sample Output

CI8

```
#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;
}
```

14. MATRIX EVEN ODD

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 sort the even and odd matrices in ascending order and print the sum of 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
Sum = 7
```

C SOLUTION

```c
#include <stdio.h>

int main ()
{
   int arr[100];
   int length, i, j, oddlen, evenlen, temp, c, d;
   int odd[50], even[50];

   printf ("enter the length of array : ");
   scanf ("%d", &length);
```

```
for (i = 0; i < length; i++)
  {
     printf ("Enter element at %d index : ", 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++)    // seperation 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; i++)    // sorting of even array
  {
     for (j = i + 1; j < evenlen; j++)
  {
```

```
temp = 0;
        if (even[i] > even[j])
        {
           temp = even[i];
           even[i] = even[j];
           even[j] = temp;
        }
  }
  }

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

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

  printf ("\n");
  printf ("Sorted odd array : ");    // printing odd array
```

```
 for (i = 0; i < oddlen; i++)
   {
      printf ("%d ", odd[i]);
   }

   printf ("\n\n%d", even[evenlen - 2] + odd[oddlen-2]);    // printing
final result
}
```

JAVA SOLUTION

```
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<Integer>ArrayList();
      ArrayList<Integer> odd = new<Integer>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));
   }
}
```

15. MATHS IS FUN

You are required to implement the following function:
int Calculate(int m, int n);

accenture
specific training

The function accepts 2 positive integers ‘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 <= 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

terv

C SOLUTION

```
#include <stdio.h>

int Calculate (int, int);
int main ()
{
   int m, n, result;
   // Getting Input
   printf ("Enter the value of m : ");
   scanf ("%d", &m);
   printf ("Enter the value of n : ");
   scanf ("%d", &n);

   result = Calculate (n, m);
   // Getting Output

   printf ("%d", result);
   return 0;
}

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

JAVA SOLUTION

```
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);
   }
}
```

16. Write a program to find the length of the longest common subsequence between two strings.

```
def longest_common_subsequence(s1, s2):
    m, n = len(s1), len(s2)
    dp = [[0] * (n + 1) for _ in range(m + 1)]

    for i in range(1, m + 1):
        for j in range(1, n + 1):
            if s1[i - 1] == s2[j - 1]:
                dp[i][j] = dp[i - 1][j - 1] + 1
            else:
                dp[i][j] = max(dp[i - 1][j], dp[i][j - 1])

    return dp[m][n]

string1 = input("Enter first string: ")
string2 = input("Enter second string: ")
print("Length of longest common subsequence:",
longest_common_subsequence(string1, string2))
```

17. Write a program to find the length of the longest palindrome subsequence in a string.

```
def longest_palindrome_subsequence(s):
    n = len(s)
    dp = [[0] * n for _ in range(n)]

    for i in range(n - 1, -1, -1):
        dp[i][i] = 1
        for j in range(i + 1, n):
            if s[i] == s[j]:
                dp[i][j] = dp[i + 1][j - 1] + 2
            else:
```

```
 dp[i][j] = max(dp[i + 1][j], dp[i][j - 1])

    return dp[0][n - 1]

string = input("Enter a string: ")
print("Length of longest palindrome subsequence:",
longest_palindrome_subsequence(string))
```

18. Write a program to find all permutations of a string.

```
from itertools import permutations

def string_permutations(s):
    return ["".join(p) for p in permutations(s)]

string = input("Enter a string: ")
print("All permutations:", string_permutations(string))
```

19. SMALEST PRODUCT

def ProductSmallestPair(sum, arr)

The function accepts an integers 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] <= 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

```
#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;
}
```

20. MOVES

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

```
#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;
}
```