DSA

Q1-Search-Array-16921

Count the number of lower case characters from the array of characters having alphanumeric symbol. The size of array should be greater than 0 and less than equal to 20. If size is not in the mention range than program should display “Invalid array size” without asking for the second input. And if no lower case character is present then display -1.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   int n;
   cin>>n;
   int lowercase=-1;
   if(n>0 && n<=20)
   {
      char arr[n];
      for(int i=0;i<n;i++)
      {
         cin>>arr[i];
      }
      for(int i=0;i<n;i++)
      {
    if(islower(arr[i]))
    {
      lowercase++;
```

```
    }
      }
      cout<<lowercase;
   }
   else
   {
      cout<<"Invalid array size";
   }
   return 0;
}
```

Q2- **Array_Deletion-1_16920**

During the get together party, Liza asked all her friends to play a game where all the N friends have been asked to arrange themselves in a row and assigned some unique number.

As per the rule of the game, Liza will show a number and the friend with that number needs to get out of the row and the vacant space should be occupied by all the friends on the right side such that the sequence is not disturbed.

Example:

If there are 5 friends with the numbers 4, 8, 3, 6, 2 and Liza has shown the number 8 then the remaining friends should have the numbers: 4, 3, 6, and 2.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   int n;
   cin>>n;
   int arr[n];
```

```
    for(int i=0;i<n;i++)
    {
        cin>>arr[i];
    }
    int n_dlt;
    cin>>n_dlt;
    for(int i=0; i<n; i++)
    {
        if(arr[i]==n_dlt)
        {
            n=n-1;
            for(int j=i; j<n; j++)
            {arr[j] = arr[j+1];}
        }
    }
    for(int j=0;j<n;j++)
    {
        cout<<arr[j]<<" ";
    }

    return 0;
}
```

Q3- **Array_Insertion_1_26199**

A task is assigned to some surveyors to store temparature in an array daily for 8 days.

An array for storing temperature is used by the surveyor but they forget to insert one element at some positions. Index 0 1 2 3 4 5 6 7 Temp. 30 40 35 25 20 10 23

If they insert 15 at index 4 then the array looks like:

Index 0 1 2 3 4 5 6 7
Temparature 30 40 35 25 15 20 10 23

So, you decided to create a function to insert the temperature at any given position in an array so, that if insertion is required then they can insert it.

```
#include<iostream>
using namespace std;
int main() {
char str[100],c;
   int x[100],n=0,i=0,temp=0,t,j=0;
   cin.getline(str,100);


   while(str[i]!='\0')
   {
      if(str[i]==' ')
      {


      x[j]=temp;
      temp=0;

      j=j+1;

      }

      else
      {
        c=str[i];
        t=(int(c)-48);
        temp=temp*10+t;
      }


     i+=1;
```

```
    }
    x[j]=temp;

    j=j+1;
    cin>>n;
    cin>>temp;



   if(n<0||n>j)
       cout<<"Invalid Position";
    else
    {
       for(i=j;i>n;i--)
          x[i]=x[i-1];

       x[i]=temp;

       for(i=0;i<j;i++)
          cout<<x[i]<<",";
       cout<<x[j];

    }

    return 0;
}
```

Q4- **Array_Traversal-1_26699**

Alice went for shopping and bought 8 goods costing different prices. At the time of billing she realises that she did not have enough money. Now, she have decided to remove the item costing maximum amount and replace it with another item whose price is same as the item costing minimum amount. Help Alice and display final prices on the screen.

Note: if more than one element cost maximum price, replace the first item.

Example 1: Input: 250 1000 50 20 10 100 200 25 Output: 250 10 50 20 10 100 200 25

Example 2: Input: 2500 2500 50 20 10 100 200 25 Output: 10 2500 50 20 10 100 200 25

```
#include <bits/stdc++.h>

using namespace std;

void replaceMax(float arr[], int n)
{

   // Maximum element from the array
   float max = *max_element(arr, arr + n);

   // Minimum element from the array
   float min = *min_element(arr, arr + n);


   // Assuming all the array elements are distinct
   // Replace the maximum element with
   // the coefficient of range of the array
   for (int i = 0; i < n; i++) {
      if (arr[i] == max) {
         arr[i] = min;
         break;
      }
   }
for(int i=0;i<8;i++){
      cout<<arr[i]<<" ";
   }
```

```
}



int main() {
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   int n=8;
   float arr[n];
   for(int i=0;i<8;i++){
      cin>>arr[i];
   }

   replaceMax(arr, n);
   return 0;
}
```

Q5- **ArrayDeletion_24906**

Today is such a beautiful evening after heavy rainfall and kids decided to play in playground, then at the same moment they are deligthed to see the colors of Rainbow in the sky. They observe seven colors like Violet, Indigo,Blue,Green,Yellow,Orange,Red i.e 'V','I','B','G','Y','O','R'.Now your task is to remove a color which is mentioned above as input and display the remaining colors of Rainbow, if the color which you want to delete that is not comes under VIBGYOR then display message as "Color not available".

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
   char arr[7]={'V','I','B','G','Y','O','R'},elem;
   int tot=7, i,  j, found=0;
```

```
    cin>>elem;
    for(i=0; i<tot; i++)
    {
        if(arr[i]==elem)
        {
            for(j=i; j<(tot-1); j++)
                arr[j] = arr[j+1];
            found++;
            i--;
            tot--;
        }
    }
    if(found==0)
        cout<<"Color not available";
    else
        for(i=0; i<tot; i++)
        cout<<arr[i]<<"";
    cout<<endl;
    return 0;
}
```

Q6- **Linear_Search_1_16856**

Given an array a[] of M elements, the task is to write a program to search a given element X in a[].

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
```

```
int main()
{
  int arr[10], tot=10, i, elem, j, found=0;
   for(i=0; i<tot; i++)
      cin>>arr[i];
   cin>>elem;
   for(i=0; i<tot; i++)
   {
      if(arr[i]==elem)
      {
         if(arr[i]==elem)
      {
         j=i+1;
         found++;
      }
      }
   }
   if(found==0)
      cout<<-1;
   else
      cout<<j<<"";
   cout<<endl;
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   return 0;
}
```

Q7- **Array_Deletion_1_27947**

Consider Aman is visiting Nehru Zoo. She has seen there are N elephants standing in a row. She wants to remove the elephants having the same height standing in consecutive.Write a program for Aman so that she can get the desired sequence of elephants.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
   int n;
   cin >> n;

   int a[n]={0};
   int b[n]={0};
   for (int i = 0; i < n; i++)
   {
      cin >> a[i];
   }
   int j=0;
   if(a[0]!=0)
   {
      b[j++]=a[0];
   }
   else
   {
      cout << "Invalid Input";
      return 0;
   }
   for(int i = 1; i < n; i++)
   {
```

```
        if(a[i]!=0)
        {
            if(a[i]!=a[i-1])
            {
                b[j++]=a[i];
            }
        }
        else
        {
            cout << "Invalid Input";
            return 0;
        }
    }
    for(int i = 0; i < j; i++)
    {
        cout << b[i] << " ";
    }

    return 0;
}
```

Q8- **LinearSearch_1_22176**

Let us assume that a child went to a shop to purchase N BROWN chocolates and while packing chocolates shopkeeper might put M white chocolate instead of brown chocolate. Find out the location of white chocolate in the box and return the location. (index of array begins from 0). In case if there is no white chocolate then print -1.Write a program for the same and implement all test cases.

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

```
using namespace std;
int main() {
   int n,m=0;
   cin>>n;
   char arr[n];
   for(int i = 0 ; i < n ; i ++)
   {
      cin>>arr[i];
   }
   for(int i = 0 ; i < n ; i ++)
   {
      if(arr[i]=='W')
      {
         cout<<i<<" ";
         m=m+1;
      }
      else
      {
         continue;
      }
   }
   if(m==0)
   {
      cout<<-1;
   }
   cout<<endl<<m;
   return 0;
}
```

Q9- **Array_Insertion_1_16915**

Consider a physical trainer is arranging the students for physical training in a school. There are N students standing in a row for physical training. He wants to insert a student with average height between the two students who are having odd height standing in consecutive. Write a program for a physical trainer so that he can get the desired sequence of students.

Example: if the 5 students are having values: 3 6 5 9 7 then 5 and 9 are the students with odd height so the student with height 7 (average of 5 and 9) should be inserted between 5 and 9. Similarly, 9 and 7 are the 2 students with odd heights so student with height 8 should be inserted between 9 and 7.

Output: 3 6 5 7 9 8 7

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
   int n,average;
   int valid = 1;
   cin>>n;
   int arr[100];
   for(int i=0;i<n;i++){
      cin>>arr[i];
   }
   for(int i=0;i<n;i++){
      if(arr[i] == 0){
         valid = 0;
      }
   }
   if(n==1 || valid == 0){
      cout<<"Invalid Input";
```

```
        return 0;
    }
    for(int i=1;i<n;i++){
        if(arr[i-1]%2!=0 && arr[i]%2!=0){
            average = (arr[i-1]+arr[i])/2;
            for(int j=n-1;j>=i;j--){
                arr[j+1] = arr[j];
            }
            arr[i] = average;
            i++;
            n++;
        }
    }
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    return 0;
}
```

Q10- **Array_Traversal-1_26108**

Assume a student is having 8 lectures on Monday of different courses listed as JAVA, Python, DBMS and DataStructures. Find out how many times JAVA lecture is repeated on Monday.

Constraints: 1. Size of the array should be 8

Output: JAVA lecture is repeating 3 times

Sample Case 1: Input: Total 8 lectures on Monday JAVA JAVA DataStructures DBMS JAVA JAVA Python DataStructures Output: JAVA lecture is repeating 4 times

Sample Case 2: Input: Total 8 lectures on Monday JAVA JAVA DataStructures DBMS JAVA JAVA Python JAVA Output: JAVA lecture is repeating 5 times

Sample Case 3: Input: Total 8 lectures on Monday Python Python DataStructures DBMS Python Python DBMS Python Output: No JAVA Lecture

```
#include <cmath>
#include <cstdio>
```

```
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   string arr[13];
for(int i =0;i<13;i++)
{
cin>>arr[i];
}
int count = 0;
for(int j=0;j<13;j++)
{
if(arr[j]=="JAVA")
{
count++;
}
}
if(count!=0)
{
cout<<"JAVA lecture is repeating "<<count<<" times";}
   else
   {
      cout<<"No JAVA Lecture";
   }
   return 0;
}
```

Q11- **Array_Deletion-2_16920**

Lewis is working on a game designing project called as Ball Blast and he is facing issue in implementing the logic for the same. There are N number of balls having a number written on each ball. All the balls are arranged in a row and as per the rules of the game, if there are 3 balls in a row having ODD number on them then the first ball of the set will be removed. The same process is to be repeated for all the balls from left to right. At the end of the game, the numbers on the remaining balls is to be printed.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int n,*ae;
void del(int temp)
{
   int i;
   for(i = temp;i<n;i++)
      {
         ae[i] = ae[i+1];
      }
}
void check()
{
   bool key = false;
   int temp,i;
   for(i = 0;i<n-2;i++)
   {
      if(ae[i] % 2 != 0 && ae[i+1] % 2 != 0 && ae[i+2] % 2 != 0)
         {
            temp = i;
```

```
            key = true;
            break;
        }
    }
    if(key == true)
      {
        del(temp);
        n--;
        check();
      }
    else return;
}
int main()
{
    int i;
    cin>>n;
    ae = new int[n];
    for(i = 0;i<n;i++)
    {
      cin >> ae[i];
    }
    check();
    for(i = 0;i<n;i++)
    {
      cout<<ae[i]<<" ";
    }
    return 0;
}
```

Q12- **Array_insertion_1_28066**

Suppose You are having 13 playing cards of Heart and among these card you have to pick n cards in sorted order. Your task is to pick another card from the remaining and insert it in proper position of the previously sorted array

example if you have picked 5 cards in sorted order 4 6 8 9 12 and then you pick next card 7 you have to insert the card 7 in its sorted position

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main(){
   int *p;
   int n;
   cin>>n;
   p=new int[n+1];
   for(int i=0;i<n;i++){
      cin>>p[i];
   }
   int x;
   cin>>x;
   int y;
   for (int i=n-1; i>=0; i--){
      if (p[i] < x) {
         y = (i+1);
         break;
      }
      else {
         y = 0;
      }
```

```
    }
    for(int i=n-1;i>=y;i--){
        p[i+1]=p[i];// shifting right side
    }
    p[y]=x;
    for(int i=0;i<n+1;i++){
        cout<<p[i]<<" ";
    }
}
```

Q13- **Traversal_Array_21482**

Akash got a new coin game. In the coin game, Akash will only win the game if the total count is 56. Each coin will have value inscribed. Write a programming solution, to help Akash in counting total value of coins at the end of the game. Coin Game is played N times.

- **Note:** Player will have exactly K coins at the end of the game.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
  int i,n;

    cin>>n;
    int a[7];
    for(i=0;i<n;i++)
    {
    int sum=0;
     for(int j=0;j<7;j++)
     {
```

```
        cin>>a[j];
        sum=sum+a[j];
     }

    if(sum==56)
        cout<<1<<endl;
    else
        cout<<0<<endl;
    }
}
```

Q14- **Linked_List_Searching-1_16920**

An IT company named as **Cloudo_Tech** is maintaining the records of its employees using a linked list where the following details of employees are maintained:

1. Name
2. Employee ID
3. Salary

The company management is looking for an interface where they can search the details of any employee using the Employee ID. You have to implement the solution for the same in the following function:

void search_employee(int emp_id)

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

using namespace std;

struct Employee
{
    string emp_name;
```

```
    int salary;
    int emp_id;
    Employee *next;
};

Employee *Cloudo_Tech = NULL;

void add_employee()
{
   //Creating Employee

    Employee *emp = new Employee;
    cin>>emp->emp_name;
    cin>>emp->emp_id;
    cin>>emp->salary;

  //Inserting Employee in the beginning of the list

    emp->next = Cloudo_Tech;
    Cloudo_Tech = emp;

}

void search_employee(int id)
{
   Employee *p= Cloudo_Tech;
    int yes=0;
    while(p!=NULL){
        if(p->emp_id == id){
            cout<<p->emp_name<<" "<<p->salary;
```

```
            yes=1;
        }
        p=p->next;
    }
    if(yes==0){
        cout<<"Employee does not exist";
    }
}


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */

    int n;
    cin>>n;     //Reading number of employees

    for(int i=0; i<n; i++)       //adding employees in the list
      add_employee();
    int item;
    cin>>item;

    search_employee(item);









    return 0;
}
```

Q15- **Insertion-Sort-2-21482**

Akash is found of collecting unique coins. Every coin has a denomination i.e Z. One Afternoon, he has K coins. He wnat to arrange K coins in non-increaasing order as per denomination. Write a programming solution to help akash. - **Note:** Use Insertion Sort

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
  //array declaration
   int n,i,j;
   int temp;
   cin>>n;
   int arr[n];
   //read n elements
   for(i=0;i<n;i++)
   {
      cin>>arr[i];
   }

   //sorting - Descending ORDER
   for(i=0;i<n;i++)
   {
      for(j=i+1;j<n;j++)
      {
         if(arr[i]<arr[j])
         {
            temp  =arr[i];
```

```
            arr[i]=arr[j];
            arr[j]=temp;
          }
       }
    }
    //print sorted array elements
    for(i=0;i<n;i++)
       cout<<arr[i]<<" ";
    cout<<endl;

    return 0;
}
```

Q16- **Selection_Sort-2_26108**

Bob considered an array of Strings and he sorted the Strings in alphabetical order

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <string>
using namespace std;
int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int N,x;
    cin>>N;
    string A[N],y;
    for(int i=0;i<N;i++){
       cin>>A[i];
    }
```

```
    for(int i=0;i<N-1;i++){
        x=i;
        for(int j=i;j<N;j++){
            if(A[j]<A[x]){
                x=j;
            }
        }
        y=A[i];
        A[i]=A[x];
        A[x]=y;
        for(int q=0;q<N;q++){
            cout<<A[q]<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

Q17- **Linked_List_Linear_Search-1_27947**

A linked list is used to store Roll Numbers of N students. Priyanka want to check that whether data is stored in palindrome format or not. Write a program to help the priyanka which will provide result in 'Yes' or 'No' depending on the data stored in list is in palindrome format.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
string make_palindrome(int num)
{
    string str=to_string(num);
```

```
    string new_str;
    int n=str.length();
    for (int i=n-1; i>=0; i--)
    {
        new_str+=str[i];
    }
    // cout<<new_str;
    return new_str;
}
int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int n;
    cin>>n;
    if (!(n>=5 and n<15))
    {
        cout<<"Invalid Input";
        return 0;
    }
    int arr[n];
    for (int i=0; i<n; i++)
    {
        cin>>arr[i];
    }
    int mid, j_trav;
    if (n%2==0)
    {
        mid=n/2;
        j_trav=mid;
    }
```

```
    else{
        mid=(n+1)/2;
        j_trav=mid-1;
    }
    int possible=1;
    for (int i=0; i<mid; i++)
    {
        for (int j=n-1; j>=j_trav; j--)
        {
            if (!(make_palindrome(arr[i])==to_string(arr[j])))
            {
                possible=0;
                break;
            }
            i++;
        }
        if (!possible)
        {
            break;
        }
    }
    if (possible)
    {
        cout<<"Yes";
        return 0;
    }
    cout<<"No";
    return 0;
}
```

Q18- **LinkedListSearching1_24906**

Surbhi's birthday is on September 1,2022 and she wants to buy a pair of shoe. As she is very excited for her special day so she decided for early shopping. That's why, she went to Mall and selected best 5 pair of shoe but there is problem is with financial condition so she selected that pair who is having minimum cost? Example 1. 23 45 67 100 2 2

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <bits/stdc++.h>
using namespace std;
struct Node {
   int data;
   struct Node* next;
};
int largestElement(struct Node* head)
{
   int max = INT_MIN;
   while (head != NULL) {
      if (max < head->data)
         max = head->data;
      head = head->next;
   }
   return max;
}
int smallestElement(struct Node* head)
{
   int min = INT_MAX;
   while (head != NULL) {
      if (min > head->data)
```

```
            min = head->data;
        head = head->next;
    }
    return min;
}
void push(struct Node** head, int data)
{
    struct Node* newNode =
        (struct Node*)malloc(sizeof(struct Node));
     // Assign the data into newNode.
    newNode->data = data;
    // newNode->next assign the address of
    // head node.
    newNode->next = (*head);

    // newNode become the headNode.
    (*head) = newNode;
}
// Driver program to test the functions
int main()
{
int arr[5];
    struct Node* head = NULL;
    for(int i=0;i<5;i++){
        cin>>arr[i];
    }
    push(&head, arr[0]);
    push(&head, arr[1]);
    push(&head, arr[2]);
```

```
    push(&head, arr[3]);
    push(&head, arr[4]);
    cout << smallestElement(head) << endl;

    return 0;
}
```

Q19- **array_bubblesort-2_26699**

In a sports event, Lindsey, a famous athlete, has decided to meet her fans and shake hands with them. The order in which she will meet them is the fan degree. i.e. the fan with maximum degree will be considered first. Now, the fans are required to take positions according to the degree value. While they are not allowed to swap positions with any other person, but with the person at their adjacent places. Write a program to implement the same.

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
void swap(int* xp, int* yp)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}
// A function to implement bubble sort
void bubbleSort(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)

        // Last i elements are already in place
        for (j = 0; j < n - i - 1; j++)
```

```
            if (arr[j] < arr[j + 1])
                swap(&arr[j], &arr[j + 1]);
}
/* Function to print an array */
void printArray(int arr[], int size)
{
   int i;
   for (i = 0; i < size; i++)
       printf("%d ", arr[i]);
   printf("\n");
}
// Driver program to test above functions
int main()
{
   int arr[] = {4,10,6,9,5,1,8,2,7,3};
   int n = sizeof(arr) / sizeof(arr[0]);
   bubbleSort(arr, n);
   printArray(arr, n);
   return 0;
}
```

Q20- **LinkedList_Traversing_1_26199**

Write a program to find the sum of odd nodes and even nodes from a linked list.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
struct node
{
```

```
int data;
node *link;
};
node *start = NULL;
node *last = NULL;
void insert(){
   node *nd =new node;
   cin>>nd->data;
   if(start == NULL)
   {
      nd->link = start;
      start =nd;
      last =nd;
   }
   else{
      nd->link = NULL;
      last->link = nd;
      last = nd;
   }
 }
int evenT=0;
int oddT=0;
void search_sums(){
    node *p=start;

   while(p != NULL){
       if((p->data)%2==0){

          evenT=p->data+evenT;
```

```
        }
        else{
            oddT=p->data+oddT;
            }

        p=p->link;

    }

    cout<<oddT<<" "<<evenT;

}

int main() {

    int n;

    cin>>n;
    if(n>=0 && n<=7){
        for(int i=0 ; i<n; i++){
            insert();
        }
        search_sums();
     }
    else{
        cout<<"Invalid Range";
    }
    return 0;
}
```

Q21- **Binary_Search_2_16856**

Take an array of integers nums of size N from user which is sorted in ascending order, and an integer target, write a program to search target in nums. If target exists, then return its index. Otherwise, return -1. You must write a program with O(log n) runtime complexity.

```
#include <iostream>

#include <algorithm>

using namespace std;


int binary_search1(int *a, int n, int item){
   int beg = 0, end = n-1, mid;
   while(beg <= end){
      mid = (beg+end)/2;
      if(item<a[mid]){
         end = mid-1;
      }
      else if(item>a[mid]){
         beg = mid+1;
      }
      else{
         return mid;
      }
   }
   return -1;
}


int main() {
   int n; cin >> n;
   if(n==1){
      cout << 1;
   }
```

```
    else {
        int a[n];
        for(int i=0; i<n; i++){
            cin >> a[i];
        }
        int item; cin >> item;
        cout << binary_search1(a, n, item);
    }
    return 0;
}
```

Q22- **Linked_List_Traversing_1_16915**

Write a program to display the count of nodes having odd number in a given singly linked list. If no odd number is present in the linked list then display "No odd number present". Linked list will contain only positive number, if you do not want add next number to the linked list then enter -1.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

struct node{
    int data;
    node *next;
};
struct node *head=NULL;

void insert(int new_data){
    struct node *new_node=new node;
    new_node->data=new_data;
```

```
    new_node->next=head;
    head=new_node;
}

void counteven(int count=0){
    struct node *ptr;
    ptr=head;
    while(ptr!=NULL){
        if(ptr->data%2!=0)
            count++;
        ptr=ptr->next;
    }
    if(count>0)
        cout<<count;
    else
        cout<<"No odd number present";
}

int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int a;
    while(true){
        cin>>a;
        if(a==-1)
            break;
        else
            insert(a);
    }
```

```
    counteven();



    return 0;
}
```

Q23- **ArrayMerging_2_22176**

Consider there are N number of employees in Programming domain sitting in one room and have their unique emp_id and M number of employees of Automata domain sitting in an another room with their unique emp_id. As per new guidelines, sitting arrangements have been updated and now all members of Automata domain are shifted in Programming domain. Now as per new sitting arrangement firstly programming domain employees will occupy the cubicle and then automata domain employee will occupy the cubicle(index begins from 0) .Also find the cubicle of employees having even emp_id.If there is no employee with even emp_id then print -1.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
    int n,m;
    cin>>n;
    int a[n];
    for(int i=0;i<n;i++)
    {
        cin>>a[i];
    }

    cin>>m;
    int b[m];
    for(int i=0;i<m;i++)
```

```
{
    cin>>b[i];
}

int c[m+n];
for(int i=0;i<m+n;i++)
{
    if(i<n)
    c[i] = a[i];
    else
    c[i] = b[i-n];
}

for(int i=0;i<m+n;i++)
{
    cout<<c[i]<<" ";
}
cout<<"\n";
bool flag=0;
for(int i=0;i<m+n;i++)
{
    if(c[i]%2==0)
       {
          flag=1;
          cout<<i<<" ";
       }
    else{
        continue;
    }
```

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



    return 0;
}
```

Q24- **Linked_List_Traversing_1_28066**

WAP to add all the even Numbers Present in the Singly Linked List. if no even numbers Present in the link list then display "No Even numbers Present".

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;



// Represents node of the linked list
struct Node {
    int data;
    Node* next;
};

// Function to insert a node at the
// end of the linked list
void insert(Node** root, int item)
{
    Node *ptr = *root, *temp = new Node;
    temp->data = item;
```

```
    temp->next = NULL;

    if (*root == NULL)
        *root = temp;
    else {
        while (ptr->next != NULL)
            ptr = ptr->next;
        ptr->next = temp;
    }
}

// Function to print the sum of even
// and odd nodes of the linked lists
void even(Node* root)
{
    int even = 0;
    bool flag=0;
    Node* ptr = root;
    while (ptr != NULL) {

        // If current node's data is even
        if (ptr->data % 2 == 0){
            even += ptr->data;
            flag=1;}
        // ptr now points to the next node
        ptr = ptr->next;
    }
    if(flag==0){cout<<"No Even numbers Present"<<endl;}
    else{cout << even << endl;}
```

```
}


// Driver code
int main()
{
   int n;
   cin>>n;
   int arr[n];
   for (int i=0;i<n;i++){
      cin>>arr[i];
   }
   Node* root = NULL;
   for(int i=0;i<n;i++){
      insert(&root, arr[i]);
   }


   even(root);


   return 0;
}
```

Q25- **merge-array2-16921**

You are given two integers representing the size of two different sized arrays. Take the elements of the arrays from the user in the non-increasing order. If the elements are not in sorted order in any array, display the message “Incorrect Array Elements”. Merge the elements of the given arrays in sorted order and display them. The size of the array should be greater than 0 and less than equal to 20. If array will not be in the given range then display the message “Invalid Array”.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
```

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

int input(int *temp ,int n){
    int check = 99999999;
    for(int i = 0 ; i < n ; i++){
        cin>>temp[i];
        if( check < temp[i] ){
            return 1;
        }
        check = temp[i];
    }
    return 0;
}
int main() {
    int n ,m;
    cin>>n;
    if( n <= 0 || 20 < n ){    cout<<"Invalid Array"; return 0;   }
    int arr[n];
    if( input(arr,n) ){
        cout<<"Incorrect Array Elements";
        return 0;
    }
    cin>>m;
    if( m <= 0 || 20 < m || n == m){    cout<<"Invalid Array"; return 0;   }
    int brr[m];
    if( input(brr,m) ){
        cout<<"Incorrect Array Elements";
        return 0;
```

```
    }
    int crr[n+m] ,i = 0 ,j = 0 ,k = 0;
    while( i < n && j < m ){
        if(arr[i] >= brr[j])
            crr[k++] = arr[i++];
        else
            crr[k++] = brr[j++];
    }
    while(i < n)
        crr[k++] = arr[i++];
    while(j < m)
        crr[k++] = brr[j++];
    for(int i = 0 ;i < k; i++)
     cout<<crr[i]<<endl;
    return 0;
}
```

Q26- **Linked_List_Traversing_1_26121**

The LPU named as **Fees_Due_Record** is maintaining the records of its student Fee details using a linked list where the following details of employees are maintained:

4. Student Registration Number
5. Name
6. Due Fee

The LPU is looking for an interface where they can search the details of student who have fee due more than 25000. You have to implement the solution for the same in the following function: **void Insert_Details(int , string, int) void Fees_Due_Details()**

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
```

```
struct node{
   int reg;
   string name;
   int due;
   node *link;
};
node *start = NULL;
node *last = NULL;

void insert(){
   node *nd =new node;
   cin>>nd->reg>>nd->name>>nd->due;
   if(start == NULL)
   {
      nd->link = start;
      start =nd;
      last =nd;
   }
   else{
      nd->link = NULL;
      last->link = nd;
      last = nd;
   }

}
void Fees_Due_Details(){
   int count=0;
   node *p=start;
   while(p!=NULL){
```

```
        if((p->due)>25000){
            count++;
           cout<<p->reg<<" "<<p->name<<" "<<p->due;
           cout<<endl;
        }

        p=p->link;
    }
    if(count == 0){
        cout<<"Students does not have fee due more than 25000";
    }

}

int main() {
    int n;
    cin>>n;
    if(n>0){
        for(int i=0; i<n; i++){
            insert();
        }
        Fees_Due_Details();
    }
    return 0;
}
```

Q27- **merge-array-16921**

You are given two integers representing the size of two different sized arrays. Take the elements of the arrays from the user in the non-decreasing order. If the elements are not in sorted order in any array, display the message “Incorrect Array Elements”. Merge the elements of the given arrays in sorted order and display them. The size of the array should

be greater than 0 and less than equal to 20. If array will not be in the given range then display the message “Invalid Array”.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
int n1,n2,small;
cin>>n1;
if(n1<=0 || n1>20){
cout<<"Invalid Array";
return 0;
}
int arr1[n1];
for(int i=0;i<n1;i++){
cin>>arr1[i];
}
small = arr1[0];
for(int i=1;i<n1;i++){
if(arr1[i]<small){
cout<<"Incorrect Array Elements";

return 0;
}
else{
small = arr1[i];
}
}
```

```
cin>>n2;
if(n1==n2){
cout<<"Invalid Array";
return 0;
}
if(n2<=0 || n2>20){
cout<<"Invalid Array";
return 0;
}
int arr2[n2];
for(int i=0;i<n2;i++){
cin>>arr2[i];
}
small = arr2[0];
for(int i=1;i<n1;i++){
if(arr2[i]<small){
cout<<"Incorrect Array Elements";
return 0;
}
else{
small = arr2[i];
}
}
int arr[n1+n2];
int i=0,j=0,c=0;
while(true){
if(i==n1 || j==n2){
break;
}
```

```
if(arr1[i]<arr2[j]){
arr[c] = arr1[i];
i++;
}
else{
arr[c] = arr2[j];
j++;
}
c++;
}
if(i!=n1){
for(int x=c;x<n1+n2;x++){
arr[x] = arr1[i];
i++;
}
}
if(j!=n2){
for(int x=c;x<n1+n2;x++){
arr[x] = arr2[j];
j++;
}
}
for(int i=0;i<n1+n2;i++){
cout<<arr[i]<<endl;
}
return 0;
}
```

Q28- **LinkedListSearching2_24906**

Ruchika's birthday is on September 1,2022 and she wants to buy a pair of shoe. As she is very excited for her special day so she decided for early shopping. That's why, she went to

Mall and selected best 5 pair of shoe but there is no problem is with financial condition so she selected that pair who is having maximum cost? Example 1. 23 45 67 100 2 100

```
#include<iostream>

using namespace std;

int main()

{

int a[5],max=0;

for (int i=0;i<5;i++)

{

cin>>a[i];

if(a[i]>max)

{

max=a[i];

}

}

cout<<max;

return 0;

}
```

Q29- **ArrayMerging_1_22176**

Consider there are N number of employees in Programming domain sitting in one room and have their unique emp_id and M number of employees of Automata domain sitting in an another room with their unique emp_id. As per new guidelines, sitting arrangements have been updated and now all members of Automata domain are shifted in Programming domain. Now as per new sitting arrangement firstly programming domain employees will occupy the cubicle and then automata domain employee will occupy the cubicle .

```
#include<iostream>

using namespace std;

int main()

{

int a[50] ;// i

int b[50] ; // j

int c[100]; //k
```

```
int i,j,k;
int asize,bsize,csize=0;

cin>>asize;

for(i=0;i<asize;i++)
cin>>a[i];

cin>>bsize;

for(j=0;j<bsize;j++)
cin>>b[j];

csize=asize+bsize;

i=0,j=0,k=0 ;

while(i<asize && j<bsize)
{
if(a[i]<b[j])
{
c[k]=a[i];
k++;
i++;
}
else
{ c[k]=b[j];
k++;
j++;
}
```

```
}
while(j<bsize)
{
c[k]=b[j];
k++;
j++;
}
while(i<asize)
{
c[k]=a[i];
k++;
i++;
}
```

```
for(k=0;k<csize;k++ )
cout<<c[k]<<"\t";
return 0;
}
```

Q30- **Linked List Insertion-1 16920**

After discussing the concepts of One-way Linked List, Professor Samuel asked his students to implement a function for inserting a node in the singly linked list such that the charcters present in the linked list nodes follow the given condition:

7. A vowel must not be present adjacent to another vowel
8. 3 consecutive consonants must not be there in the list

Students were supposed to implement a function to insert the node with given charater in the very first suitable position of the list.

Example: **C -> B -> E -> K -> A**

Now If the character X is to be inserted in the list then, it must be inserted between E and K.

C -> B -> E -> X -> K -> A

In the above example, if A is to be inserted then, It can be inserted in the beginning only.

A -> C -> B -> E -> K -> A

If there is no suitable position available then print **Can't be inserted**

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int che=0;
class node
{

   char data;
   node *next=NULL;
   public:
   void input()
   {
      cin>>data;
   }
   void link(node *p)
   {
      next = p;
   }
   void check(char c)
   {
      if(c =='A' || c == 'E' || c == 'I' || c == 'O' || c =='U')
      {
         che=1;
```

```
    }
    else
    {
       che=0;
    }
}
void in(char inserted,int c)
{
    int no=0;

    node *start=this,*cur=start,*pre=NULL;
    node *p=NULL;
    if(c==1)
    {

       while(cur!=NULL)
       {
          check(cur->data);
          if(che==0)
          {
             if(cur==start)
             {
                start = new node;
                start->data=inserted;
                start->next=cur;
                no=1;
                break;
             }
             else
```

```
    {
        check(pre->data);
        if(che==0)
        {
            p = new node;
            p->data=inserted;
            pre->next=p;
            p->next=cur;
            no=1;
            break;
        }
        else
        {
        if(cur->next!=NULL)
        {
        check(cur->next->data);
        if(che==0)
        {
            p = new node;
            p->data=inserted;
            p->next=cur->next;
            cur->next=p;
            no=1;
            break;
        }
        }
        }
    }
}
```

```
        pre=cur;
        cur=cur->next;
    }
 }

else if(c==0)
 {
    check(start->data);
    if(che==1)
    {
            start = new node;
             start->data=inserted;
             start->next=cur;
             no=1;
    }
    else
    {
     while(cur!=NULL)
    {
        check(cur->data);
        if(che==1)
        {
           if(cur->next!=NULL)
             {check(cur->next->data);
              if(che==1)
              {
                 p = new node;
                 p->data=inserted;
                 p->next=cur->next;
```

```
    cur->next=p;
    no=1;
    break;
}
else
{
  if(cur->next->next!=NULL)
    {
       check(cur->next->next->data);
       if(che==1)
       {
           p = new node;
           p->data=inserted;
          p->next=cur->next;
            cur->next=p;
            no=1;
            break;
       }
    }
    else
    {
       p = new node;
       p->data=inserted;
       p->next=cur->next;
       cur->next=p;
       no=1;
       break;
    }
```

```
            }
            }
          else
          {
           p= new node;
           p->data=inserted;
           cur->next=p;
           no=1;
           break;
          }
          }
      pre=cur;
      cur=cur->next;
   }
   }

}

if(no==0)
{
   cout<<"Can't be inserted";
}
else
{
   p=start;
while(p!=NULL)
{
   cout<<p->data<<" ";
   p=p->next;
```

```
        }
        }
    }

};

int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    char insert;
    int check = 0;
    node *start=NULL,*cur=NULL,*pre=NULL;
    int n;
    cin>>n;
    for(int i=0;i<n;i++)
    {
        if(start==NULL)
        {
            start=new node;
            start->input();
            pre=start;
        }
        else{
            cur=new node;
            cur->input();
            pre->link(cur);
            pre=cur;
        }
    }
    cin>>insert;
```

```
   if(insert =='A' || insert == 'E' || insert == 'I' || insert == 'O' || insert =='U')
   {
     check =1;

   }

   start->in(insert,check);
   return 0;
}
```

Q31- **Circular-Linklist-2-21482**

Ayushi got a new problem in her subject. She was given a Circular Singly Linklist in which every node contains a pointer ‘next’ which points to the next node in the list. Last node ‘next’ pointer points to the first node in the linked list. Every node has a ‘data’ part that contains some integer positive. You have to delete the node in the linked list at the given position N.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

class node
{
 int data;
 node *next=NULL;
 public:
 void input()
 {
   cin>>data;
 }
```

```
 void link(node *p)
 {
  next =p;
 }
  void task(int n)
  {
     node *start=this;
     node *pre=NULL;
   node *cur=start;
   for(int i=0;i<n;i++)
   {
      pre=cur;
      cur=cur->next;
   }
   pre->next= cur->next;
   cur=start;
     cout<<cur->data;
     cur=cur->next;
     while(cur!=start)
     {
        cout<<" "<<cur->data;
        cur=cur->next;
     }

  }
};
node *start=NULL;
int main() {
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
```

```
node *pre=NULL,*cur=NULL;
int to_delete;
cin>>to_delete;
int n;
cin>>n;
for(int i=0;i<n;i++)
{
   if(start==NULL)
   {
      start = new node;
      start->input();
      pre=start;
   }
   else if(i==n-1)
   {
      cur =new node;
      cur->input();
      pre->link(cur);
      cur->link(start);
   }
   else
   {
      cur=new node;
      cur->input();
      pre->link(cur);
      pre=cur;
   }
}
start->task(to_delete);
```

```
    return 0;
}
```

Q32- **CircularLinkedList_2_26108**

Aman is playing a game where he has integer numbers as the data in each node and the data in each node is distinct. He traverses the whole list but prints only first occurrence of even numbers in the list. He traverses the whole list till he prints all the even numbers in the list

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

struct node
{
    int data;
    node *next=NULL;
    void input()
    {
        cin>>data;
    }
    void link(node *p)
    {
        next = p;
    }
};
node *start = NULL;
node *pre=NULL, *cur=start;
void even_check(int x)
{
    int a[x];
    int count=0;
    cur=NULL;
    while(cur!=start)
```

```
    {
        if(cur==NULL)
        {
            cur = start;
        }
        if((cur->data)%2==0)
        {

            a[count]=cur->data;
            count++;
            for(int i=0;i<count;i++) cout<<a[i]<<" ";
            cout<<endl;
        }
        cur=cur->next;
    }
}

int main()
{

    int n;
    cin>>n;
    for(int i=0;i<n;i++)
    {
        if(start==NULL)
        {
            start=new node;
            start->input();
            pre=start;
```

```
        }
        else
        {
            cur=new node;
            cur->input();
            pre->link(cur);
            if(i==n-1)
            {
                cur->next=start;
            }
            pre=cur;
        }
    }
    even_check(n);
}
```

Q33- **Doubly_Insertion-2-26699**

After an announcement made in a lecture room, it was announced that the students are required to reach auditorium to attend a session. While the auditorium is already almost full and have random free seats, the students of the section decided that each one of them will be holding the address of the next student and the previous student. Means, each student here is aware about the seating of the two other students except the first and the last student to enter the auditorium, who are not aware of the previous and the next student respectively. Write a program to send 10 students to auditorium while each node stores roll number in the data part.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
```

```
int main() {
int arr[10];
for(int i=0;i<10;i++)
{
cin>>arr[i];
}
for(int i=0;i<10;i++)
{
cout<<arr[i]<<" ";
}
return 0;
}
```

Q34- **LinkedListInsertion-1_24906**

On September 1,2022 Data Structure exam was taken and it was written by 5 students, and accordingly concerned faculty member uploaded the marks of those 5 students .But one of the student was not able to attempt exam so with the request , his exam was re-conducted and later on faculty is going to upload marks ,but before this faculty must needs to enter rollno and then after his marks.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;



int main() {
int arr[6];
for(int i=0;i<5;i++)
{
cin>>arr[i];
```

```
}
int pos,num;
cin>>pos;
cin>>num;
for(int i=5;i>pos-1;i--)
{
arr[i]=arr[i-1];
}
arr[pos-1]=num;
for(int i=0;i<6;i++)
{
cout<<arr[i]<<" ";
}
return 0;
}
```

Q35- **delete_node_linked_list_1_26199**

Write a program to delete the node from given position in the linked list. The list may be empty after you delete the node. In that case, print "SLL is Empty". If position of linked list is not valid then print a message of "Invalid Position"

Use the following structure to implement it.

```
struct node SLLNode
{
int data;
SLLNode* next;
};
#include <cmath>

#include <cstdio>

#include <vector>
```

```
#include <iostream>

#include <algorithm>

using namespace std;

struct node{

    int data;

    node *link;

};

node *start = NULL;

node *last = NULL;

void insert(){

    node *nd =new node;

    cin>>nd->data;

    if(start == NULL)

    {

        nd->link = start;
```

```
        start =nd;

        last =nd;

    }

    else{

        nd->link = NULL;

        last->link = nd;

        last = nd;

    }



}

void print(int n){

    node *p=start;

    int pos;

    cin>>pos;

    int x=0;
```

```
    if(n<pos+1 || pos<0){
            cout<<"Invalid Position";
    }else if(n!=1 || pos!=0){
        while(p!=NULL){
            if(pos!=x){
                cout<<p->data<<" ";
            }
            p=p->link;
            x++;
        }
    }else{
        cout<<"SLL is Empty";
    }

}
```

```
int main() {

    int n;

    cin>>n;

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

        insert();

    }

    print(n);

    return 0;

}
```

Q36- **TWOWAY_LIST_2_16856**

Given the pointer to the head node of a doubly linked list, reverse the order of the nodes in place. That is, change the next and prev pointers of the nodes so that the direction of the list is reversed. Return a reference to the head node of the reversed list. Note: The head node might be NULL to indicate that the list is empty.

```
#include <bits/stdc++.h>

using namespace std;

class DoublyLinkedListNode {

    public:

        int data;
```

```
        DoublyLinkedListNode *next;

        DoublyLinkedListNode *prev;

        DoublyLinkedListNode(int node_data) {

            this->data = node_data;

            this->next = nullptr;

            this->prev = nullptr;

        }

};

class DoublyLinkedList {

    public:

        DoublyLinkedListNode *head;

        DoublyLinkedListNode *tail;

        DoublyLinkedList() {

            this->head = nullptr;

            this->tail = nullptr;
```

```
        }

        void insert_node(int node_data) {

            DoublyLinkedListNode* node = new DoublyLinkedListNode(node_data);

            if (!this->head) {

                this->head = node;

            } else {

                this->tail->next = node;

                node->prev = this->tail;

            }

            this->tail = node;

        }

};

void print_doubly_linked_list(DoublyLinkedListNode* node, string sep, ofstream& fout) {

    while (node) {

        fout << node->data;
```

```
        node = node->next;

        if (node) {

            fout << sep;

        }

    }

}

void free_doubly_linked_list(DoublyLinkedListNode* node) {

    while (node) {

        DoublyLinkedListNode* temp = node;

        node = node->next;

        free(temp);

    }

}

// Complete the reverse function below.

/*
```

```
* For your reference:
*
* DoublyLinkedListNode {
*     int data;
*     DoublyLinkedListNode* next;
*     DoublyLinkedListNode* prev;
* };
*
*/
DoublyLinkedListNode* reverse(DoublyLinkedListNode* head)
{
    // Complete this function
    // Do not write the main method.
    DoublyLinkedListNode *current = head;
    DoublyLinkedListNode *temp = NULL;
```

```
    while ( current != NULL) {

        temp = current -> prev;

         current -> prev =  current -> next;

         current -> next = temp;

        current =  current -> prev;



    }



    if (temp != NULL)

        head = temp -> prev;

    return head;

}

int main()

{
```

```
ofstream fout(getenv("OUTPUT_PATH"));

int t;

cin >> t;

cin.ignore(numeric_limits<streamsize>::max(), '\n');

for (int t_itr = 0; t_itr < t; t_itr++) {

    DoublyLinkedList* llist = new DoublyLinkedList();

    int llist_count;

    cin >> llist_count;

    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    for (int i = 0; i < llist_count; i++) {

        int llist_item;

        cin >> llist_item;

        cin.ignore(numeric_limits<streamsize>::max(), '\n');

        llist->insert_node(llist_item);

    }
```

```
        DoublyLinkedListNode* llist1 = reverse(llist->head);

        print_doubly_linked_list(llist1, " ", fout);

        fout << "\n";

        free_doubly_linked_list(llist1);

    }

    fout.close();

    return 0;

}
```

Q37- **Linked_List_Deletion_1_16915**

Write a program to delete all nodes having odd number in a given singly linked list. If no odd number is present in the linked list then display "No odd number present". Linked list will contain only positive number, if you do not want add next number to the linked list then enter -1.

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

struct node
{
    int data;
    struct node *next;
};
```

```
struct node *head=NULL;

void insert(int d)
{
   struct node *temp;
   struct node *newnd=malloc(sizeof(struct node));
   newnd->data=d;
   newnd->next=NULL;
   if(head==NULL)
   {
      head=newnd;
   }
   else
   {
      for(temp=head;temp->next!=NULL;temp=temp->next);
      temp->next=newnd;
   }
}
void delete_at(int place)
{
   struct node *temp=head;
   int i;
   if(place==1)
   {
      head=NULL;
   }
   for(i=1;i<place-1;i++,temp=temp->next);
   if(temp->next!=NULL)
```

```
    {
        temp->next=(temp->next)->next;
    }
}

int deleteOdd()
{
    int count=0;
    int place=1;
    struct node *temp=head;
    if(head==NULL)
    {
        //this will never occur as given constraint
    }
    else
    {
        do
        {
        if(temp->data%2!=0)
        {
            delete_at(place);
            count++;
            place--;
        }
        place++;
        temp=temp->next;
        }while(temp!=NULL);
    }
   return count;
```

```
}

void display()
{
   struct node *temp=head;
   if(head==NULL)
   {
      //this will never occur as given constraint
   }
   else
   {
      do
      {
      printf("%d ",temp->data);
      temp=temp->next;
      }while(temp!=NULL);
   }
}




int main() {
   int oddcount=0;
   int n;
   while(1)
   {
      scanf("%d",&n);
      if(n==-1)
         break;
```

```
        insert(n);
    }
    oddcount=deleteOdd();
    if(oddcount==0)
        printf("No odd number present");
    else
        display();
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Q38- **DoublyLinkedList_Deletion_2_22176**

Given a doubly linked list containing N nodes. In N nodes the data part represents the roll no's of students. The task is to delete all the odd roll no's from the list.Create a program to insert N rollno's from the tail of doubly linked list,

Input:
5 15 <=> 16 <=> 6 <=> 7 <=> 17 Output: 16 <=> 6 Explanation: 15,7 and 17 are odd rollno's. So we need to delete them.

Input: 4 6 <=> 8 <=> 4 <=> 2 Output: No student is having odd roll number.

Input: 18 Wrong Input.

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

struct node
{
    struct node *front;
    int data;
    struct node *back;
};
struct node *head=NULL;
```

```
void insert(int d)
{
   struct node *temp,*newnd;
   newnd=(struct node*)malloc(sizeof(struct node));
   newnd->data=d;
   newnd->front=newnd->back=NULL;

   if(head==NULL)
   {
      head=newnd;
   }
   else
   {
      for(temp=head;temp->back!=NULL;temp=temp->back);
      temp->back=newnd;
      newnd->front=temp;
   }
}

void display()
{
   struct node *temp=head;
   if(head==NULL)
      printf("No student is having odd roll number.");
   else
   {
   while(temp!=NULL)
   {
      printf("%d ",temp->data);
```

```
        temp=temp->back;
    }
}}

void delete_at(int place)
{
    struct node *temp;
    int i;
    if(place==1)
    {
        temp=head;
        head=temp->back;
    }
    else
    {
        for(temp=head,i=1;i<place;i++,temp=temp->back);
        temp->front->back=temp->back;
        if(temp->back!=NULL)
            temp->back->front=temp->front;
      //  printf("%d,",temp->data);
    }
    free(temp);
}

int deleteOdd()
{
    int count=0,place=1;
    struct node *temp=head;
    while(temp!=NULL)
```

```
    {
        if(temp->data%2!=0)
        {
            delete_at(place);
            place--;
            count++;
        }
        place++;
        temp=temp->back;
    }
    return count;
}


int main() {
    int n,a,oddCount=0;
    scanf("%d",&n);
    if(n>=3 && n<12)
    {
        while(n!=0)
        {
            scanf("%d",&a);
            insert(a);
            n--;
        }
      //  display();
        oddCount=deleteOdd();
        if(oddCount==0)
        {
```

```
            printf("No student is having odd roll number.");
        }
        else
            display();
    }
    else
        printf("Wrong Input.");

     return 0;

}
```

Q39- **Linked_List_Deletion_1_26121**

Rahul asked his friend to write a program to delete all nodes having vowels (a, e, i, o, u) character in a given singly linked list. Where each node store a character. If no vowels found in linked list then display "No vowel in linked list". In Linked list each node contain one character. The node will be defined :

class Node { public: char data; Node* next; }

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

class node{
    public:
    char data;
    node *next;
};
class process{
    node *head;
```

```
public:
process(){
    head=NULL;
}
void insert(char d){

    node *n=new node;
    n->data=d;
    n->next=NULL;
    if(head==NULL){
        head=n;

    }
    else{
        node *temp=head;
        while(temp->next!=NULL){
            temp=temp->next;
        }
        temp->next=n;

    }
}
void Do(){
    node *temp=head;
    int x=0,q=0;
    while(temp!=NULL){
        if(temp->data=='a'||temp->data=='e'||temp->data=='i'||temp->data=='o'||temp-
>data=='u'){
            x++;
        }
```

```
        temp=temp->next;
      }
      x--;
      //cout<<x<<endl;
      if(x==0){
        cout<<"No vowel in linked list";
      }
      else{
        temp=head;
        while(temp!=NULL){
          if(temp->data=='a'||temp->data=='e'||temp->data=='i'||temp->data=='o'||temp->data=='u'){
            x++;
          }
          else{
            if(q!=0)cout<<"-->";
            cout<<temp->data;
            q++;

          }
          temp=temp->next;
        }
      }
    }

};

int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    process l;
```

```
    char a='a',b='b';
    while(a!=b){
        b=a;
        cin>>a;
        l.insert(b);
    }
    l.Do();
    return 0;
}
```

Q40- **TwoWayLinkedListDeletion1_16921**

Create a two-way linked list of N integer elements. Delete all the occurrences of the elements from the first occurrence till second last occurrence of the entered specific value. The number of elements should be greater than 2 and less than equal to 20 in the list. If size is not in the mention range than program should display “Invalid list size” without asking for the second input. If the number to be deleted is not present or present only one time, display the message “Deletion not possible”.

```
#include <iostream>

using namespace std;
class node
{
public:
    int val;
    node *prev;
    node *next;
};

class Linkedlist
{
public:
    node *first;
```

```
node *last;
Linkedlist();
void create();
void control(int);
void traversal_back();
void forward()
{
    node *temp = first;
    while (temp != NULL)
    {
        cout << temp->val << endl;
        temp = temp->next;
    }
}
void del(node *iter)
{
    node *del = iter;
    // for (int i = 0; i < io; i++)
    // {
    //     del = del->next;
    // }
    if (del == first)
    {
        first = del->next;
    }
    if (del == last)
    {
        last = del->prev;
    }
```

```
        if (del->next != NULL)
            del->next->prev = del->prev;

        if (del->prev != NULL)
            del->prev->next = del->next;
        delete del;
    }
    int ocor(int key)
    {
        node *pemp = first;
        int occur = 0;
        while (pemp != NULL)
        {
            if (pemp->val == key)
            {
                occur = occur + 1;
            }
            pemp = pemp->next;
        }
        return occur;
    }
};

Linkedlist::Linkedlist()
{
    first = NULL;
    last = NULL;
}
```

```
void Linkedlist::create()
{
   int t;
   node *temp;
   temp = new node;
   cin >> t;
   temp->val = t;
   temp->next = NULL;
   temp->prev = NULL;
   if (first == NULL)
   {
      first = temp;
      last = temp;
   }
   else
   {
      last->next = temp;
      temp->prev = last;
      last = temp;
   }
}

void Linkedlist::traversal_back()
{
   node *temp = last;
   if (temp != NULL)
   {
      while (temp != NULL)
      {
```

```
            cout << temp->val << endl;
            temp = temp->prev;
        }
        // cout << "NULL";
    }
    else
    {
        printf("Linked unserflow");
    }
}

void Linkedlist::control(int key)
{
    node *pemp = first;
    bool dalg = false;
    int occur = ocor(key);

    while (pemp != NULL)
    {
        if (pemp->val == key)
        {
            if (occur != 1)
            {
                del(pemp);
                dalg = true;
                occur = occur - 1;
            }
        }
        pemp = pemp->next;
```

```
    }
    if (dalg == true)
    {
        traversal_back();
    }
    else
    {
        cout << "Deletion not possible" << endl;
    }
}
int main()
{
    Linkedlist l;
    int N;
    cin >> N;
    if (N <= 20 && N > 2)
    {
        for (int i = 0; i < N; i++)
        {
            l.create();
        }
        int key;
        cin >> key;
        l.control(key);
    }
    else
    {
        printf("Invalid list size");
    }
```

```
    return 0;
}
```

Q41- **LinkedListInsertion-2_24906**

On September 5,2022 Python exam was taken and it was written by 6 students, and accordingly concerned faculty member uploaded the marks of those 6 students .But one of the student was not able to attempt exam so with the request , his exam was re-conducted and later on faculty is going to upload marks ,but before this faculty must needs to enter rollno and then after his marks.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;



int main() {
    int arr[6];
    int lab = 0;
    int pos,n;
    for(int i=0;i<6;i++){
        cin>>arr[i];
    }
    cin>>pos;
    cin>>n;
    for(int i=0;i<6;i++){
        if(i+1 == pos && lab == 0){
            lab = 1;
            cout<<n<<" ";
            i--;
        }
```

```
      else{
         cout<<arr[i]<<" ";
      }
   }
   return 0;
}
```

Q42- **DoublyLinkedList_Deletion_1_22176**

Create a node with a name as Node. A Node object has an integer data field, and a Node forwd pointer, pointing to forward node (i.e.: the next node in a list) and a Node back pointer ,pointing to previous node(i.e,: the previous node in a list). In this node builder is inserting N house numbers at the end of list. A node is used to store N House Numbers .Create a program to help a builder to remove the last house number from colony and then print the list in both forward and backward direction.If N>=5 && N<15 condition is not satisfied then print Invalid Input.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
class node{
  public:
  node *pre;
  node *next;
  int data;
};
class link{
   public:
   void creat();
   node *head,*tail;
   void del();
   void bshow();
```

```
    void fshow();
    link();
};
link::link()
{
    head=tail=NULL;
}
void link::creat()
{
    node *new_node=new node();
    cin>>new_node->data;
    new_node->next=NULL;
    if(head==NULL)
    {
        head=new_node;
        new_node->pre=NULL;
        tail=new_node;
    }
    else{
        node *temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }

        temp->next=new_node;
        new_node->pre=temp;
        tail=new_node;
    }
```

```
}
void link::fshow()
{
   node *temp;
   temp=head;
   while(temp!=NULL)
   {
      cout<<temp->data<<" ";
      temp=temp->next;
   }
   cout<<endl;
}
void link::bshow()
{
   node *temp;
   temp=tail;
   while(temp!=NULL)
   {
      cout<<temp->data<<" ";
      temp=temp->pre;
   }
}
void link::del()
{
   node *temp;
   temp=tail;
   tail=tail->pre;
   tail->next=NULL;
   delete temp;
```

```
}
int main() {
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   link l;
   int max;
   cin>>max;
   if(max>=5 && max<15)
   {
      for(int i=0;i<max;i++)
      {
         l.creat();
      }

      l.del();
      l.fshow();
      l.bshow();
   }
   else
   {
      cout<<"Invalid Input";
   }

   return 0;
}
```

Q43- **Queue using Two Stacks**

A queue is an abstract data type that maintains the order in which elements were added to it, allowing the oldest elements to be removed from the front and new elements to be added to the rear. This is called a *First-In-First-Out* (FIFO) data structure because the first element added to the queue (i.e., the one that has been waiting the longest) is always the first one to be removed.

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


struct node{
   int data;
   struct node* next;
};


struct queue{
   struct node* head1;
   struct node* head2;
};


struct node* push(struct node* temp,int data){
   struct node* newnode = (struct node*)malloc(sizeof(struct node));
   newnode->data = data;
   newnode->next = temp;
   temp = newnode;
   return temp;
}


void enqueue(struct queue* q,int data){
   q->head1 = push(q->head1,data);
}
```

```
int pop(struct node** temp){
   int data = ( *temp )->data;
   (*temp) = (*temp)->next;
   return data;
}


void dequeue(struct queue*q){
   int x;
   if(q->head2 == NULL){
      while(q->head1){
         x = pop( &(q->head1) );
         q->head2 = push(q->head2,x);
      }
      x = pop(&(q->head2));
   }else{
      x = pop(&(q->head2));
   }
}


void print(struct queue* q){
   int x;
   if(q->head2){
   }else{
      while(q->head1){
         x = pop((&q->head1));
         q->head2 = push(q->head2,x);
      }

   }
```

```
    x = q->head2->data;
    printf("%d\n",x);
}


int main(void){
    int data,queries,choice;
    struct queue* q  = (struct queue*)malloc(sizeof(struct queue));
    q->head1 = q->head2 = NULL;
    scanf("%d",&queries);
    for(int i = 0 ;i < queries ;i++){
        scanf("%d",&choice);
        if(choice == 1){
            scanf("%d",&data);
            enqueue(q,data);
        }else if(choice == 2){
            dequeue(q);
        }else if(choice == 3){
            print(q);
        }
    }

}
```

Q44- **Queue_2_26108**

Given an integer K and a queue of integers, we need to reverse the order of the first K elements of the queue, leaving the other elements in the same relative order.

```
#include <bits/stdc++.h>

using namespace std;


void reverseQueueFirstKElements(int k,int N , queue<int>& Queue)
```

```
{
    if (Queue.empty() == true || k > N)
        return;
    if (k <= 0)
        return;

    stack<int> Stack;


    for (int i = 0; i < k; i++) {
        Stack.push(Queue.front());
        Queue.pop();
    }

    while (!Stack.empty()) {
        Queue.push(Stack.top());
        Stack.pop();
    }


    for (int i = 0; i <N- k; i++) {
        Queue.push(Queue.front());
        Queue.pop();
    }
}


void Print(queue<int>& Queue)
{
    while (!Queue.empty()) {
```

```
        cout << Queue.front() << " ";
        Queue.pop();
    }
}

// Driver code
int main()
{
    queue<int> Queue;
    int N;cin>>N;
    int a;
    for(int i=0;i<N;i++){
        cin>>a;
        Queue.push(a);
    }

    int k;
    cin>>k;
    reverseQueueFirstKElements(k,N, Queue);
    Print(Queue);
}
```

Q45- **Stack-1-27947**

Ravi and Amit are playing a game where Ravi is going to give n strings to Amit and Amit need to tell Yes or No for each string. Yes if string is a Palindrome and No if it is not. Write a program using stack to help the Amit to find string is Palindrome or not.

```
#include <bits/stdc++.h>
using namespace std;



int main() {
```

```
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int T;
    string A,B;
    cin>>T;
    if(T<1){
       cout<<"Invalid Input";
       return 0;
    }
    while(T--){
       cin>>A;
       B=A;
       reverse(B.begin(),B.end());
       if(A==B)cout<<"Yes"<<endl;
       else cout<<"No"<<endl;
    }
    return 0;
}
```

Q46- **Balanced Brackets**

A bracket is considered to be any one of the following characters: (,), {, }, [, or].

Two brackets are considered to be a *matched pair* if the an opening bracket (i.e., (, [, or {) occurs to the left of a closing bracket (i.e.,),], or }) *of the exact same type*. There are three types of matched pairs of brackets: [ ], { }, and ( ).

A matching pair of brackets is *not balanced* if the set of brackets it encloses are not matched. For example, { [( ]) } is not balanced because the contents in between { and } are not balanced. The pair of square brackets encloses a single, unbalanced opening bracket, (, and the pair of parentheses encloses a single, unbalanced closing square bracket,].

```
#include <cmath>
#include <cstdio>
#include <stack>
#include <iostream>
#include <algorithm>
```

```
using namespace std;



int main() {

    int t,i;
    string s;
    stack<char>a;
    cin>>t;
    while(t--){
        cin>>s;
        for(i=0;i<s.size();i++){
            if(s[i]=='('
                a.push(s[i]);
            else if(s[i]=='{')
                a.push(s[i]);
            else if(s[i]=='['){
                a.push(s[i]);
            }
            else if(s[i]==')'){
                if(!a.empty()){
                if(a.top()=='('
                    a.pop();
                else
                    break;
                }
                else
                    break;
            }
```

```
    else if(s[i]=='}'){
        if(!a.empty()){
            if(a.top()=='{')
            a.pop();
        else
            break;
        }
        else
            break;
    }
    else if(s[i]==']'){
        if(!a.empty()){
        if(a.top()=='[')
            a.pop();
        else
            break;
        }
        else
            break;
    }
}

if(a.empty()&&i==s.size())
    cout<<"YES"<<endl;
else
    cout<<"NO"<<endl;
while(!a.empty()){

    a.pop();
```

```
    }
  }
  return 0;
}
```

Q47- **Queue_1_22176**

Write a program to create the queue of N elements . Size of the queue should be greater than 2 and less than equal to 15. If the size of the queue will not be in the mentioned range then it should display the message “Invalid Queue range" with taking any elements for input.And if there is no even element present then display "No even element is there".

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int n;
int *queue;
int front=-1,rear=-1;
void enqueue(int x)
{
  if(front==-1)
  {
    front=0;
    rear=0;
  }
  else if(rear==n-1)
  {
    rear=0;
  }
  else
  {
```

```
        rear++;
    }
    queue[rear]=x;
}
int dequeue()
{
    int deleted=queue[front];
    if(front ==n-1)
    {
        front=0;
    }
    else{
        front=front+1;
    }
    return deleted;
}
int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */

     int check=0;
    cin>>n;
    if(n>2 && n<=15)
    {
        queue=new int[n];
        for(int i=0;i<n;i++)
        {
            int x;
            cin>>x;
            enqueue(x);
```

```
        }
        for(int i=0;i<n;i++)
        {

            int y=dequeue();
            if(y%2==0)
            {
                cout<<y<<" ";
                check=1;
            }
        }
        if(check==0)
        {
            cout<<"No even element is there";
        }
    }
    else
    {
        cout<<"Invalid Queue range";
    }
    return 0;
}
```

Q48- **stack1_26199**

Mr. Rohan packed his books in a box having capacity to put 6 books. Now he wants to arrange the books in an another box. He is picking one book at at time and then putting it in another box. Write a program that will display the books he popped out and inserte in another box(reverse order). Create push and pop functions to implement it. If the first box is empty then display message of "Empty Box". If the number of box is not capable of storing the books then display a message of "Full Box".

```
#include <iostream>
#include <stack>
```

```
using namespace std;
int main()
{
   stack<string> newstack;
   int n;
   cin>>n;

   for(int k=0;k<n;k++){

      string p;
      cin>>p;
      newstack.push(p);}


   if(newstack.empty ()){cout<<"Empty Box";}
      else
      {while (!newstack.empty () )
      {
         if(n<=6){
         for(int i=0;i<n;i++){
         std::cout<< newstack.top ();
         if(i<n-1){cout<<"->";}
         newstack.pop();  }}
         else{cout<<"Full Box";
            return 0;}
      }}
      return 0;
}
```

Q49- **Queue-2-21482**
Implement a Queue as mentioned:

An opertaion Z is of 2 Types.

9. 1 (opertion of this type means: delete the element from the queue and insert that element again in the queue)
10. 0 (opertion of this type means you need to delete in queue)

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <queue>
using namespace std;

int main() {
   int n ,m;
   cin>>n>>m;
   queue<int> qu;
   for(int i = 0 ; i < n ; i++){
      int temp;
      cin>>temp;
      qu.push(temp);
   }
   while(m--){
      cin>>n;
      if( n == 1 )
         qu.push(qu.front());
      qu.pop();
   }
   while(!qu.empty())
      cout<<qu.front()<<" " ,qu.pop();
   return 0;
```

}

Q50- **queue_2_22176**

Assume 9 spectators are standing in a ticket counter to take a ticket to watch a T20 match.In a queue enter n names of spectators and then some gifts will be given to odd ticket number spectators who came to watch match.Note index is ticket number and values in queues are name of spectators.Create a user interactive program where 1 will be for insertion and 2 will display names of odd ticket spectators who will get some gifts.

```
#include <bits/stdc++.h>

using namespace std;



int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    vector<string>A;
    int T=0,N;
    string X;
    while(T!=2){
        cin>>T;
        if(T==1){
            cin>>N;
            while(N--){
                if(A.size()<9){
                    cin>>X;
                    A.push_back(X);
                }
                else{
                    cout<<"No other spectator is allowed";
                    return 0;
                }

            }
```

```
        }
        if(T==2){
           for(int i=0;i<A.size();i++){
              if(i%2!=0)cout<<A[i]<<" ";
           }
        }
    }
    return 0;
}
```

Q51- **Stack_1_16915**

Construct a stack using two queues (q1, q2), you need to simulate the stack operations by using queue operations. You are required to complete the three methods push() which takes an integer 'x' as input denoting the element to be pushed into the stack, pop() which poped out from the stack and display() which display the resultant stack(-1 if the stack is empty).

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
int stack[100],n=100,top = -1;
void push(int d){

        top++;
        stack[top]= d;

}
void pop(){

        top--;
```

```
}
void display(){
   if(top>=0){
   for(int i=0;i<=top;i++){
      cout<<stack[i]<<" ";
   }
   }
   else{
      cout<<-1;
   }
}
int main() {
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   int a,el;
   do{
      cin>>a;
   switch(a){
      case 1:
         cin>>el;
         push(el);
         break;
      case 2:
         pop();
         break;
      case 3:
         display() ;
         break;
   }
```

```
    }while(a!=3);
    return 0;
}
```

Q52- **Stack1_28066**

WAP to insert N Elements into the stack. Find the Middle element , then delete one element from the stack and again find the new middle element

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int *stack =NULL;
int n;
int top =-1;

void push(int x)
{
    stack[++top]=x;
}
void pop()
{
    if(top>-1)
    {
        top--;
    }
}
void mid()
{
```

```
    int mid = top/2;
    if(mid!=-1)
    {
        cout<<stack[mid]<<endl;
    }
}
int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    cin>>n;
    if(n>=3)
    {
        stack = new int[n];
        for(int i=0;i<n;i++)
        {
            int x;
            cin>>x;
            push(x);
        }
        mid();
        pop();
        mid();
    }
    return 0;
}
```

Q53- **QueueImplementation2_16921**

Write a program to implement the queue of characters in the array of size 7 in circular order. Take 2 variables ‘front’ and ‘rear’ contains the indexed of the first and last elements of the queue. Where the ‘-1’ in front and rear will represent the empty queue. Your program should be user interactive where 1 will be for insertion and 2 will be for deletion of elements. When the user will enter any other option, it should display the value of the front and rear.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

char queue[7];
int front =-1, rear=-1;
int c=1;
void enqueue(char x)
{
   if((front==0 && rear==6) || front==rear+1)
     { cout<<"OVERFLOW\n";
       c=0;
     }

   else
   {
      if(front==-1)
      {
         front =0;
         rear =0;
      }
      else if(rear==6)
         rear=0;
      else
         rear++;
```

```
        queue[rear]=x;
    }
}
void dequeue()
{
    if(front!=-1)
    {
        if(front==rear)
        {
            front=-1;
            rear=-1;
        }
        else if(front==6)
            front=0;
        else
            front++;
    }
}

int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int choice;
    do
    {
        cin>>choice;
        switch(choice)
        {
            case 1:
                {
```

```
            char x;
            cin>>x;
            enqueue(x);

            break;
          }
        case 2:
          {
            dequeue();
            break;
          }
        default:
          {
            cout<<queue[front]<<endl;
            cout<<queue[rear];
            exit(0);
          }
      }
      if(c==0)
      {
        exit(0);
      }
    }
    while(true);
    return 0;
}
```

Q54- **STACK_APPLICATION_1_26121**

Radha asked to her friend can you verify that given expression have balanced parenthesis or not, So her decided to write a code to examine that the given an string expression whether have the pairs of parenthesis are balanced or not. The orders of “(“, “)”, “[“, “]” ,“{“, “}”,are correct in the given expression.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <stack>
using namespace std;

bool areBracketsBalanced(string expr)
{
    stack<char> temp;
    for (int i = 0; i < expr.length(); i++) {
        if (temp.empty()) {
            temp.push(expr[i]);
        }
        else if ((temp.top() == '(' && expr[i] == ')')
                || (temp.top() == '{' && expr[i] == '}')
                || (temp.top() == '[' && expr[i] == ']')) {

            temp.pop();
        }
        else {
            temp.push(expr[i]);
        }
    }
    if (temp.empty()) {

        return true;
    }
```

```
    return false;
}
int main()
{
    string expr;
    cin>>expr;

    if (areBracketsBalanced(expr))
        cout << "Balanced";
    else
        cout << "Not Balanced";
    return 0;
}
```

Q55- **QueueImplementation1_16921**

Write a program to create the queue of N elements in increasing order. Size of the queue should be greater than 2 and less than equal to 10. All the elements of the queue should be in increasing order. Element will not inserted in the queue if it will not follow the above condition.

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    cin>>n;
    if(n<=2 || n>10)
    {
        cout<<"Invalid size";
        return 0;
    }
    queue<int> q;
    int count=0;
```

```
int data;
for(int i=0; i<10; i++)
{

   if(i ==0)
   {
      cin>>data;
      q.push(data);
      count++;
   }
   cin>>data;
   if(data<q.front())
   {
      continue;
   }
   else if(data>q.front())
   {
      q.push(data);
      count++;
   }
   if(count==n)
   {
      break;
   }

}
while(!q.empty())
{
   cout<<q.front()<<endl;
```

```
        q.pop();
    }
    return 0;
}
```

Q56- **Stack2_24906**

2 Ridhima visited super market to purchase fruits, and there all the fruits were displayed,then she got confused that which among the all fruit she should select or reject. All the fruits are fresh hence she decided to purchase more than one fruit but not all of them because that may reach her out of budget. So let's help Ridhima by creating one application which execute as per her choice for example when she enter 1 then she should be allowed to purchase any fruit, when she enter 2 then only recent fruit which is added that will be removed and display the remaining fruits and after that application will close. Note:-Ridhima can enter her choice repeatedly.

```
#include <cmath>

#include <cstdio>

#include <vector>

#include <iostream>

#include <algorithm>

using namespace std;

struct stack

{

  string fruit;

    stack *next=NULL;
```

```
    stack *pre=NULL;
};
stack *top=NULL;
void push()
{
    int n;
    cin>>n;
    for(int i=0;i<n;i++)
    {
        if(top==NULL)
        {
            top= new stack;
            cin>>top->fruit;
        }
        else
```

```
        {
            stack *p = top;
            top= new stack;
            cin>>top->fruit;
            top->pre = p;
            p->next=top;
        }
    }
}
void pop()
{
    cout<<top->fruit<<endl;
    top=top->pre;
    top->next=NULL;
    stack *p=top;
```

```
    while(p!=NULL)
    {
        cout<<p->fruit<<" ";
        p=p->pre;
    }
}
int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int choice;
    do
    {
        cin>>choice;
        switch(choice)
        {
            case 1: push();break;
```

```
        case 2: pop();exit(0);

      }

   }

   while(true);

   return 0;

}
```

Q57- **Queue-1-21482**

Implement a Queue as mentioned:

An operation Z is of 2 Types.

11. 1 p (operation of this type means you need to add p in queue)
12. 0 (operation of this type means you need to pop in queue and print the popped element)

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include<queue>
using namespace std;
queue<int> task;
int main() {
   /* Enter your code here. Read input from STDIN. Print output to STDOUT */
   int n,choice;
   cin>>n;
   for (int i=0;i<n;i++)
```

```
  {
    cin>>choice;
    if(choice==1)
    {
      int x;
      cin>>x;
      task.push(x);
    }
    else if(choice==0)
    {


        int y = task.front();
        cout<<y<<" ";

      task.pop();
    }

  }
  return 0;
}
```

Q58- **Queue_1_26699**

A movie cinema has limited seats for the viewers. The cinema is distributing tickets to people and keeping a record of their names. Taking array index as seat number(first seat is at number 0), store the names of people keeping in mind that after all the seats are full, “Houseful” should be displayed, otherwise "Booked" should be displayed.

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

```
using namespace std;



int main() {
   int num,ns;
   string name;
   cin >> num;
   cin >> ns;
   if(ns > num)
      cout << "Houseful";
   else
   {
      while(ns > 0){
      cin >> name;
         ns--;
      }
      cout << "Booked";
   }
   return 0;
}
```

Q59- **Stack-2-27947**

Chanden is using a compiler which is able to solve postfix expressions only. Postfix expressions are those expressions in which operator comes after the operands. Chanden is not able to understand how compiler solves these postfix expresssion. Write a program using stack which will work like compiler so that Chanden will be able to understand the functionality of compiler.

```
#include<bits/stdc++.h>

using namespace std;

int k=0;

float scanNum(char ch){

  int value;
```

```
  value = ch;
  return float(value-'0');
}
int isOperator(char ch){
  if(ch == '+'|| ch == '-'|| ch == '*'|| ch == '/' || ch == '^')
    return 1;
  return -1;
  }
  int isOperand(char ch){
    if(ch >= '0' && ch <= '9')
      return 1;
    return -1;
  }
  float operation(int a, int b, char op){
    //Perform operation
    if(op == '+')
      return b+a;
    else if(op == '-')
      return b-a;
    else if(op == '*')
      return b*a;
    else if(op == '/')
      return b/a;
    else if(op == '^')
      return pow(b,a);
    else
  return INT_MIN;
}
float postfixEval(char postfix[]){
```

```
int a, b;
stack<float> stk;

for(int it=0;it<k;it++){
  if(isOperator(postfix[it]) != -1){
    if(stk.empty())
    {
      return 100;
       break;
    }
    a = stk.top();
    stk.pop();
     if(stk.empty())
    {
      return 100;
        break;
    }


    b = stk.top();
    stk.pop();

    stk.push(operation(a, b, postfix[it]));
  }else if(isOperand(postfix[it]) > 0){

    stk.push(scanNum(postfix[it]));
  }
}
```

```
  return stk.top();
}
int main(){
  char post[50];
  for(int i=0;i<50;i++)
  { k++;
    cin>>post[i];
     if(post[i]<0)
     {
        break;
     }
  }

  float ans=postfixEval(post);
  if(ans!=100)
  {
    cout<<ans;
  }
  else
  {
    cout<<"Invalid Input";
  }
  return 0;
}
```

Q60- **Stack2_28066**

You have three stacks of cylinders where each cylinder has the same diameter, but they may vary in height. You can change the height of a stack by removing and discarding its topmost cylinder any number of times.

Find the maximum possible height of the stacks such that all of the stacks are exactly the same height. This means you must remove zero or more cylinders from the top of zero or more of the three stacks until they are all the same height, then return the height.

```
#include "bits/stdc++.h"
using namespace std;
#define rep(i,n) for(int (i)=0;(i)<(int)(n);++(i))
#define rer(i,l,u) for(int (i)=(int)(l);(i)<=(int)(u);++(i))
#define reu(i,l,u) for(int (i)=(int)(l);(i)<(int)(u);++(i))
static const int INF = 0x3f3f3f3f; static const long long INFL = 0x3f3f3f3f3f3f3f3fLL;
typedef vector<int> vi; typedef pair<int, int> pii; typedef vector<pair<int, int> > vpii; typedef long long ll;
template<typename T, typename U> static void amin(T &x, U y) { if(y < x) x = y; }
template<typename T, typename U> static void amax(T &x, U y) { if(x < y) x = y; }

int main() {
  int n1; int n2; int n3;
  while(~scanf("%d%d%d", &n1, &n2, &n3)) {
    vector<int> a(n1);
    for(int i = 0; i < n1; ++ i)
      scanf("%d", &a[i]);
    vector<int> b(n2);
    for(int i = 0; i < n2; ++ i)
      scanf("%d", &b[i]);
    vector<int> c(n3);
    for(int i = 0; i < n3; ++ i)
      scanf("%d", &c[i]);
    reverse(a.begin(), a.end());
    reverse(b.begin(), b.end());
    reverse(c.begin(), c.end());
    map<int, int> t;
    rep(k, 3) {
      const vi &v = k == 0 ? a : k == 1 ? b : c;
      int sum = 0;
```

```
            for(int x : v) {
                sum += x;
                ++ t[sum];
            }
        }
        int ans = 0;
        for(auto p : t) if(p.second == 3)
            amax(ans, p.first);
        printf("%d\n", ans);
    }
    return 0;
}
```

Q61- **Queue_1_22176**

Write a program to create the queue of N elements . Size of the queue should be greater than 2 and less than equal to 15. If the size of the queue will not be in the mentioned range then it should display the message “Invalid Queue range" with taking any elements for input.And if there is no even element present then display "No even element is there".

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <queue>
using namespace std;



int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int n;
    cin>>n;
    if(n>2 && n<=15)
```

```
{
    queue<int> q;
    int check=-1,x;
    for(int i=0;i<n;i++)
    {
        cin>>x;
        if(x%2==0)
        {
            check =1;
        }
        q.push(x);
    }
    if(check==-1)
    {
        cout<<"No even element is there";
    }
    else
    {
        for(int i=0;i<n;i++)
        {
            int y= q.front();
            if(y%2==0)
            {
                cout<<y<<" ";
            }
            q.pop();
        }

    }
```

```
    }
    else
    {
        cout<<"Invalid Queue range";
    }
    return 0;
}
```

Q62- **stack2_26199**

Write a program to create a stack for characters with push, pop functions having maximum capacity of 10 then you have to create a function to check whether given expression of '(' and ')' characters is balanced or not.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <stack>
using namespace std;

bool areBracketsBalanced(string expr)
{
    stack<char> temp;
    for (int i = 0; i < expr.length(); i++) {
        if (temp.empty()) {
            temp.push(expr[i]);
        }
        else if ((temp.top() == '(' && expr[i] == ')')
                || (temp.top() == '{' && expr[i] == '}')
                || (temp.top() == '[' && expr[i] == ']')) {
```

```
            temp.pop();
        }
        else {
            temp.push(expr[i]);
        }
    }
    if (temp.empty()) {

        return true;
    }
    return false;
}
int main()
{
    string expr;
    cin>>expr;
    if(expr.length()>10){
        cout<<"Stack Full";
        return 0;
    }

    if (areBracketsBalanced(expr))
        cout << "Balanced";
    else
        cout << "Not Balanced";
    return 0;
}
```

Q63- **Queue_1_26108**

In F1 race competition, a total of 5 cars participated and they have to complete a total of 3 laps. During the end of each lap the first car is pushed to the last and finally find the winner who comes first at the end of the 3rd lap

```
#include<iostream>
#include<queue>
using namespace std;
void display_queue(queue<string> q);
int main(){
   queue<string> q;
   string name;
   int size;
   cin>>size;
   for(int i=0;i<5;i++){
      cin>>name;
      q.push(name);
   }
   for(int j=0;j<3;j++){
      string s = q.front();
      q.pop();
      q.push(s);
      display_queue(q);
   }
   cout<<q.front();

}
void display_queue(queue<string> q) {
  while(!q.empty()) {
   cout << q.front() << " ";
   q.pop();
```

```
 }
 cout << endl;
}
```

Q64- **Stack_2_16915**

Write a program to reverse the string using stack.

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <stack>
#include <string>
using namespace std;


int main() {
string n;
getline(cin , n);
stack<char> stk;
int x = n.length();
if( x < 5 || x > 30){
return 0;
}
for( int i=0 ; i < x ; i++ ){
char v;
v=n[i];
stk.push(v);
}
while (!stk.empty()) {
cout << stk.top();
```

```
stk.pop();

}

return 0;

}
```

Q65- **QUEUE_1_16856**

All the employee (except the Head) are standing in queue to submit their bills. The employee have different seniority. In all there are N employee of K different seniority levels. These are given to you in an array, where A1, A2, ..., AN denote the seniority of employees in the queue. AN denotes front of the queue and A1 denotes end of the queue. Head gets an interesting thought past his head. He begins to think what if every employee starting from the end of the queue begins to delegate his job of submitting bills to a employee least ahead of him in the queue but junior to him. The Head’s fearfulness of this scenario is f = i2 - i1 + 1, where i1 is the index of employee in queue and i2 is the index of the junior employee. Head's total fearfulness of chaos is the product of all the fearfulness in Head’s mind. Note if a employee has no junior ahead of him/her in the queue then Head’s fearfulness for this employee is 1. You are required to find the Head's total fearfulness given an arrangement of employees in a queue. Since this number can be quite large output it modulo 1000000007.

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

long long int n,k;

cin>>n>>k;


long long int a[n];

stack <long long int>s;


int count = 0;


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

cin>>a[i];


long long int answer = 1;
```

```
for(long long int i=0;i<n;i++)
{
while(s.empty()==false && a[s.top()] > a[i]){
answer = (answer *(i-s.top() +1))%1000000007;
s.pop();
count = 1;
}
s.push(i);
}


if(count)
cout<<answer<<endl;
else
cout<<0<<endl;
return 0;
}
```