**1 . You are given an array arr of size N. You need to push the elements of the array into a stack and then print them while popping.**

#include <iostream>

#include <stack>

using *namespace* std ;

stack<*int*> mypush(*int* *arr*[], *int* *n* )

{

  // write down code

   stack<*int*> s ;

   for (*int* i = 0 ;i < n;i++)

   {

    s.push(arr[i]);

   }

   return s;

}

*void* mypop(stack<*int*> *s*)

{

   // write down code

   if(s.empty()){

    cout<<"UNDERFLOW"<<"\n";

   }

   else{

    while(!s.empty()){

*int* num = s.top() ;

       cout<<num<<" ";

       s.pop();

    }

   }

}

*int* main(){

*int* t ;

    cin >> t ;

    while(t--)

    {

*int* n ;

        cin >> n ;

*int* arr[n];

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

        cin >>  arr[i] ;

        stack<*int*> mys = mypush(arr,n);

        mypop(mys);

        cout<<endl;

    }

}

**Input:**

**1**

**5**

**1 2 3 4 5**

**Output:**

**1**

**5**

**5 4 3 2 1**

**Input:**

**1**

**7**

**1 6 43 1 2 0 5**

**Output:**

**5 0 2 1 43 6 1**

**2.Operations on Stack**

**Given a stack of integers and Q queries. The task is to perform the operation on stack according to the query.**

**The queries can be of 4 types:**

**i x: (adds element x in the stack).**

**r: (removes the topmost element from the stack).**

**h: Prints the topmost element.**

**f y: (check if the element y is present or not in the stack). Print "Yes" if present, else "No".**

**Example 1:**

**Input:**

**1**

**6**

**I 2**

**I 4**

**i 3**

**I 5**

**H**

**f, 8**

**Output:**

**5**

**No**

**Explanation:**

**Inserting 2, 4, 3, and 5 onto the stack. Returning top element which is 5. Finding 8 will give No,**

**as 8 is not in the stack.**

**Example 2:**

**Input:**

**1**

**4**

**I 3**

**I 4**

**r**

**f 3**

**Output:**

**Yes**

**Explanation:**

**Inserting 3 and 4 onto the stack. Finding 3 will give Yes as output because 3 is available in the stack.**

#include <iostream>

#include <stack>

using *namespace* std ;

*void* insert(stack<*int*> &*s*,*int* *x*){

    s.push(x);

}

*void* remove(stack<*int*> &*s*){

    s.pop();

}

//Function to print the top element of stack.

*void* headOf\_Stack(stack<*int*> &*s*){

//write code

if (s.empty())

{

    cout<<"Empty stack";

}

else{

*int* head  = s.top();

  cout<<head<<"\n";

}

}

*bool* find(stack<*int*> *s*, *int* *val*){

//write code

  if(s.empty())

  {

    cout<<"stack is empty"<<endl;

  }

  else{

       while(!s.empty())

       {

          if (s.top()==val)

          {

            return true;

          }

          else{

            s.pop();

          }

       }

  }

    return false;

}

*int* main() {

*int* t;

    cin>>t;

    while(t--)

    {

        stack<*int*> s;

*int* q;

        cin>>q;

        while(q--){

*char* ch;

            cin>>ch;

            if(ch=='i')

            {

*int* x;

                cin>>x;

                insert(s,x);

            }

            else if(ch=='r')

            {

                remove(s);

            }

            else if(ch=='h')

            {

                headOf\_Stack(s);

            }

            else if(ch=='f')

            {

*int* x;

                cin>>x;

                if(find(s,x))

                cout << "Yes";

                else cout << "No";

                cout << endl;

            }

        }

    }

    return 0;

}

**3.Reverse Array Using Stack**

**You are given an array arr[] of size N, the task is to reverse the array elements in-place by using a stack.**

**Example 1:**

**Input: N = 5, arr[] = {1, 2, 3, 4, 5}**

**Output: 5 4 3 2 1**

**Explanation: After the reverse, array will**

**look like {5, 4, 3, 2, 1}.**

**Example 2:**

**Input: N = 1, arr[] = {1}**

**Output: 1**

**Explanation: After the reverse, array will look like {1}.**

#include <iostream>

#include <stack>

using *namespace* std ;

*class* Solution{

*public:*

*void* reverseArray(*int* *n*, *int*\* *arr*){

//write code

         stack<*int*> s ;

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

         {

            s.push(arr[i]);

         }

*int* k = 0 ;

         while(!s.empty())

         {

*int* num  = s.top() ;

             arr[k++]=num;

             s.pop();

         }

    }

};

*int* main(){

*int* t;

    cin>>t;

    while(t--)

    {

*int* n;

    cin >> n;

*int* arr[n];

    for(*int* i = 0;i<n;i++){

        cin >> arr[i];

    }

    Solution obj;

    obj.reverseArray(n, arr);

    for(*int* i = 0;i<n;i++){

        cout << arr[i] << " ";

    }

    cout << endl;

    }

    return 0;

}

4. **Removing consecutive duplicates**

**You are given string str. You need to remove the consecutive duplicates from the given string using a Stack.**

**Input:**

**aaaaaabaabccccccc**

**Output:**

**ababc**

**Explanation:**

**The order is in the following way 6->a, 1->b, 2->a, 1->b, 7->c.**

**So, only one element from each group will remain and rest all are removed.**

**Therefore, final string will be:- ababc.**

**Input:**

**abbccbcd**

**Output:**

**abcbcd**

**Explanation:**

**The order is in the following way 1->a, 2->b, 2->c, 1->b, 1->c, 1->d.**

**So, only one element from each group will remain and rest all are removed.**

**Therefore, final string will be:- abcbcd.**

#include <iostream>

#include <stack>

using *namespace* std;

*class* Solution

{

*public:*

    string removeConsecutiveDuplicates(string *s*)

    {

//write code

            string ans="" ;

*int* k=-1;

            stack<*char*> mys;

            stack<*char*> temp ;

            for (*int* i  = 0 ;  i < s.length() ; i++)

            {

                mys.push(s[i]);

            }

            while(!mys.empty())

            {

                if (temp.empty() || temp.top() != mys.top())

                {

                    temp.push(mys.top());

                }

                mys.pop();

            }

            while(!temp.empty())

            {

                ans.push\_back(temp.top());

                temp.pop();

            }

            return ans;

     }

};

*int* main() {

*int* t;

    cin>>t;

    while(t--)

    {

        string s;

        cin>>s;

        Solution obj;

        cout<<obj.removeConsecutiveDuplicates(s)<<endl;

    }

    return 0;

}

--------------------------------------------------------------------------------------------------------------------------------------

5. **Removing Consecutive Elements**

**Given a vector V of N elements and two special numbers x and y. Remove all consecutive same special elements. The final vector should be free from any consecutive same special elements.**

**Note: The final vector may be empty.**

**Example:**

**Input:**

**1**

**6**

**2 1 2 2 2 5**

**1 2**

**Output:**

**2 1 2 5**

**Explanation:**

**Testcase : x = 1 and y = 2.**

**Now, we traverse the vector from left to right and remove all consecutive 1s and 2s.**

**First, we remove 2 1 2 2 2 5.**

**Now we are left with 2 1 2 5.**

**Now,there is no consecutive 1 or 2.**

**We are left with 2 1 2 5.**

#include <bits/stdc++.h>

using *namespace* std;

vector<*int*> remove\_special\_consecutive(vector<*int*>*v*,*int* *x*,*int* *y*)

{

//write code

 vector <*int*>  ans ;

 stack<*int*> mys;

            stack<*int*> temp ;

            for (*int* i  = 0 ;  i < v.size() ; i++)

            {

                mys.push(v[i]);

            }

            while(!mys.empty())

            {

                if (temp.empty() ||temp.top() != mys.top() )

                {

                    temp.push(mys.top());

                }

                mys.pop();

            }

            while(!temp.empty())

            {

                ans.push\_back(temp.top());

                temp.pop();

            }

            return ans;

}

*int* main() {

*int* t;

    cin>>t;

    while(t--)

    {

*int* n;

        cin>>n;

        vector<*int*>v;

        for(*int* i=0;i<n;i++)

        {

*int* x;

            cin>>x;

            v.push\_back(x); //Input vector elements

        }

*int* x,y;

        cin>>x>>y;

        vector<*int*>result= remove\_special\_consecutive(v,x,y);

        for(*auto* i:result)

        {

            cout<<i<<" ";

        }

        cout<<endl;

    }

    return 0;

}

-------------------------------------------------------------------------------------------------------------------------------------

**6. Get min at pop**

**You are given an array A of size N. You need to first push the elements of the array into a stack and then print minimum in the stack at each pop until stack becomes empty.**

**Input:**

**N = 5**

**A = {1 2 3 4 5}**

**Output:**

**1 1 1 1 1**

**Explanation:**

**After pushing elements to the stack,**

**the stack will be "top -> 5, 4, 3, 2, 1".**

**Now, start popping elements from the stack**

**popping 5: min in the stack is 1.popped 5**

**popping 4: min in the stack is 1. popped 4**

**popping 3: min in the stack is 1. popped 3**

**popping 2: min in the stack is 1. popped 2**

**popping 1: min in the stack is 1. popped 1**

**Example 2:**

**Input:**

**N = 7**

**A = {1 6 43 1 2 0 5}**

**Output:**

**0 0 1 1 1 1 1**

**Explanation:**

**After pushing the elements to the stack,**

**the stack will be 5->0->2->1->43->6->1.**

**Now, poping the elements from the stack:**

**popping 5: min in the stack is 0. popped 5**

**popping 0: min in the stack is 0. popped 0**

**popping 2: min in the stack is 1. popped 2**

**popping 1: min in the stack is 1. popped 1**

**popping 43: min in the stack is 1. popped 43**

**popping 6: min in the stack is 1. popped 6**

**popping 1: min in the stack is 1. popped 1.**