Lecture 28 Stack

1. Stack STL
2. #include <iostream>
3. #include <stack>
4. using namespace std;
5. int main() {
6. stack<int> s;
7. for(int i=1; i<=5; i++)
8. {
9. s.push(i);
10. }
11. while(!s.empty())
12. {
13. cout<<s.top()<<", ";
14. s.pop();
15. }
16. return 0;
17. }

2. <https://leetcode.com/problems/valid-parentheses/>

class Solution {

public:

bool isValid(string s) {

stack<char> st;

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

{

if(s[i]=='(' || s[i]=='{' || s[i]=='[')

{

st.push(s[i]);

}

else if(s[i]=='}')

{

if(st.empty() || st.top()!='{')

{

return false;

}

st.pop();

}

else if(s[i]==')')

{

if(st.empty() || st.top()!='(')

{

return false;

}

st.pop();

}

else if(s[i]==']')

{

if(st.empty() || st.top()!='[')

{

return false;

}

st.pop();

}

}

return st.empty();

}

};

3. Duplicate Parenthesis

#include <iostream>

#include <stack>

using namespace std;

bool checkExp(string str)

{

    stack<char> st;

    for(int i=0; i<str.size(); i++)

    {

        if(str[i]!=')')

        {

            st.push(str[i]);

        }

        if(str[i]==')')

        {

            if(st.top()=='(')

            {

                return false;

            }

            while(st.top()!='(')

            {

                st.pop();

            }

            st.pop();

        }

    }

    return true;

}

int main() {

    string exp;

    cin>>exp;

    if(checkExp(exp))

    {

        cout<<"No Duplicates";

    }

    else

    {

        cout<<"Duplicates";

    }

    return 0;

}

4. <https://leetcode.com/problems/min-stack/>

class MinStack {

public:

/\*\* initialize your data structure here. \*/

stack<pair<int, int>> st;

MinStack() {

}

void push(int x) {

if(!st.empty())

{

pair<int, int> t = st.top();

int minVal = t.second;

if(x < minVal)

{

minVal = x;

}

st.push(make\_pair(x, minVal));

}

else

{

st.push(make\_pair(x, x));

}

}

void pop() {

st.pop();

}

int top() {

return st.top().first;

}

int getMin() {

return st.top().second;

}

};

5. Next Larger Element

#include <iostream>

#include <stack>

using namespace std;

void largestVal(int arr[], int n, int ans[])

{

    stack<int> s;

    for(int i=0; i<n; i++)

    {

        while(!s.empty() && arr[s.top()] < arr[i])

        {

            ans[s.top()] = arr[i];

            s.pop();

        }

        s.push(i);

    }

    for(int i=0; i<n; i++)

    {

        cout<<ans[i]<<", ";

    }

}

int main() {

    int n;

    cin>>n;

    int arr[n];

    int ans[n];

    for(int i=0; i<n; i++)

    {

        cin>>arr[i];

        ans[i]=-1;

    }

    largestVal(arr, n, ans);

    return 0;

}