***Week – 6 (24.05.2021 – 29.05.2021)***

***RANDOM CODES***

1. ***Add Binary:***

class Solution {

public:

string addBinary(string a, string b) {

int i = a.size()-1, j = b.size()-1, carry = 0, sum=0;

string res="";

while(i>=0 || j>=0)

{

sum = carry;

carry = 0;

if(i>=0)

{

sum = sum + (a[i] - 48);

i--;

}

if(j>=0)

{

sum = sum + (b[j] - 48);

j--;

}

carry = sum/2;

sum = sum%2;

res = (char)(sum + 48) + res;

}

if(carry != 0) res = (char)(carry + 48) + res;

return res;

}

};

1. ***Evaluate Reverse Polish Notation:***

class Solution {

public:

string oper(int n1, int n2, string op)

{

if(op == "+") return to\_string(n2+n1);

if(op == "-") return to\_string(n2-n1);

if(op == "\*") return to\_string(n2\*n1);

if(op == "/") return to\_string(n2/n1);

return "";

}

int evalRPN(vector<string>& tokens) {

stack<string> val;

int n1, n2, i;

string n;

for(i=0; i<tokens.size(); i++)

{

if(tokens[i]=="+" || tokens[i]=="-" || tokens[i]=="\*" || tokens[i]=="/")

{

n1 = stoi(val.top());

val.pop();

n2 = stoi(val.top());

val.pop();

n = oper(n1, n2, tokens[i]);

val.push(n);

}

else val.push(tokens[i]);

}

return stoi(val.top());

}

};

1. ***Ransom Note:***

class Solution {

public:

bool canConstruct(string ransomNote, string magazine) {

vector<int> v(26,0);

for(char c : magazine)

v[c - 'a']++;

for(char c : ransomNote)

{

if(v[c-'a']==0)

return false;

v[c-'a']--;

}

return true;

}

};

1. ***Check If Word Is Valid After Substitutions:***

class Solution {

public:

bool isValid(string s) {

int i;

stack<char> st;

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

{

if(s[i] == 'a') st.push(s[i]);

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

{

if(st.empty()) return false;

if(st.top() == 'a') st.pop();

else return false;

st.push(s[i]);

}

else

{

if(st.empty()) return false;

if(st.top() == 'b') st.pop();

else return false;

}

}

if(st.empty()) return true;

else return false;

}

};

1. ***Simplify Path:***

class Solution {

public:

string simplifyPath(string path) {

stack<string> st;

string s = "", res = "";

int i;

for(i=0; i<path.size(); i++)

{

if(path[i] == '/') continue;

s = "";

while(i < path.size() && path[i] != '/')

{

s = s + path[i];

i++;

}

if(s == "..")

{

if(!st.empty())

st.pop();

}

else if(s == ".") continue;

else st.push(s);

}

while(!st.empty())

{

res = "/" + st.top() + res;

st.pop();

}

if(res == "") res = "/";

return res;

}

};

1. ***Remove Outermost Parentheses:***

class Solution {

public:

string removeOuterParentheses(string s) {

stack<char> st;

int i;

string res = "";

char temp;

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

{

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

{

if(!st.empty()) res = res + "(";

st.push(s[i]);

}

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

{

temp = st.top();

st.pop();

if(st.empty()) continue;

else res = res + ")";

}

}

return res;

}

};

1. ***Make the String Great:***

class Solution {

public:

string makeGood(string s) {

stack<char> st;

int i;

string res = "";

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

{

if(!st.empty() && islower(st.top()) && toupper(st.top()) == s[i]) st.pop();

else if(!st.empty() && isupper(st.top()) && tolower(st.top()) == s[i]) st.pop();

else st.push(s[i]);

}

while(!st.empty())

{

res = st.top() + res;

st.pop();

}

return res;

}

};

1. ***Baseball Game:***

class Solution {

public:

int calPoints(vector<string>& ops) {

int i, n1, n2, sum=0;

stack<int> s;

for(i=0; i<ops.size(); i++)

{

if(ops[i] == "C") s.pop();

else if(ops[i] == "D") s.push(2 \* s.top());

else if(ops[i] == "+")

{

n1 = s.top();

s.pop();

n2 = s.top();

s.push(n1);

s.push(n1 + n2);

}

else s.push(stoi(ops[i]));

}

while(!s.empty())

{

sum = sum + s.top();

s.pop();

}

return sum;

}

};

1. Crawler Log Folder:

class Solution {

public:

int minOperations(vector<string>& logs) {

stack<string> s;

int i, t=0;

for(i=0; i<logs.size(); i++)

{

if(logs[i] == "./") continue;

else if(logs[i] == "../" && !s.empty()) s.pop();

else if(logs[i] == "../" && s.empty()) continue;

else s.push(logs[i]);

}

while(!s.empty())

{

t++;

s.pop();

}

return t;

}

};