LC → 1021

Remove outer parenthesis

I/p → "(()())(())"

→ decomposition ⇒ "(()())" + "(())"

after removal becomes

"()() + ()"

→ So, answer ⇒ ()()() ,

Intution :

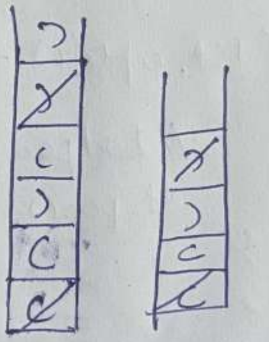
→ Check for the s.top()

if Sameas(ch) pop the element ;

and push the "Ch"

→ "(()())(())"

```
{
  if(st.top()==ch)
  {
    st.pop();
  }
  st.push(ch);
}
```



→ ()()() Hence

O/P

→ Since adjacency is to checked

LIFO is the best used/ comfortable to use

"(())(())(()())"

→ ()()()()(())

```
stack<char> st;
 strig ans = "";
for(char ch: s)
  {
   if(ch == "(")
     {
      if(!st.empty())      // not outer // pehle s hai
        {
         ans = ans + ch;
        }
         st.push(ch);
     }
   else {   // ab closing agya
      st.pop();
      if(!st.empty())  //outer
        {
         ans = ans + ch;
        }
      }
   return ans;
  }
};
```

→ (()())

→

~~(~~
(

ans = ()(
(

LC → 682 → Baseball Game

["5" "2" "C" "D" "+"]

→ ⑤ ②

(+)

→ 5+2 ⇒ ⑦

C → ~~②~~ (D)

st.push(2*st.top());

10
~~2~~
5

["5", "-2", "4", "C", "D", "9", "+", "+"]

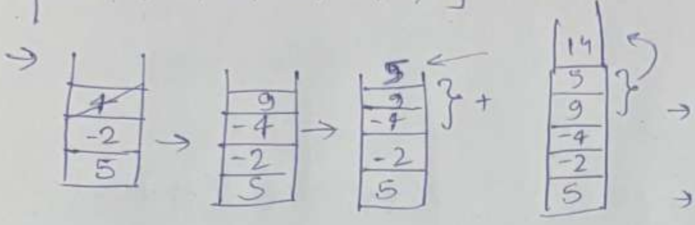
| 5, | -2 | ~~4~~ |

| 5 | -2 | ~~4~~ | 9 |
| | | 5 | |

sum = 4+9
⇒ ~~13~~
3
8
8 -

| 5 | 3 |

[5, -2, 4, C, D, 9, +, +]



→ 14 + 5 + 9 + (-4) + (-2) +5

→ 19 + 9 + 5 - (6)

→ $\boxed{27}$

```
it sum=0;
Stack<it> st;
for(auto c: operations)
{
  if (c!="+" && c!="D" && c!="C")
        // mtlb integer hai;
        it x =Stoi(c);
        st.push(x);
  }
 else if (c=='+')
   {
      a = st.top()
      st.pop();
     b = st.top()
      st.pop()
     st.push(b)          // phele (b) kyuki wo nikla phele
    st.push(a)
    st.push(a+b);
  }
 else if (C=="D")
   { it x = st.top();
     st.push(x*2);
   }
 else { st.pop();
  }
```

```
// Stack is ready, Sumup Kardo
while(!st.empty())
 {
    sum= sum + st.top()
  } st.pop();
retu sum;
 }
}
```

735- Asteriod Collision

asteriods = [5, 10, -5]

→ (5) would not collide alone

→ (10) will not they move right

→ if negative comes (-5), it will colloide with (10)
but since 10 is greater so (-5) will burst ;

→ [10, 2, -5] → 10, will go
2 will go
-5 will go.
(2 will destroy) so [10, -5] answer,

So, III cases:
(if (+) comes, it will go inside)

(I) if negative (-).

```
→ while (!s.empty () & s.top() < i && i<0)
    {  s.pop();          // remove all smaller
                            than absalute
    }
```

(II) if negative but greater is s.top()
→ asteriod destroys

(III) if equal, both destroys

LC → 71, simplify path

String begins with "/"

. → current dir

.. → previous / current directory

/// or // → /

directory names are valid

eg → ... / .. //..

banana Kya hai.

→ Start from '/'.

→ Directories must be distinguished by '/'

→ unless root '/' not at end.

→ . / .. not

eg → "/home/" → "/home"

→ "/home//foo/ → "/home/foo"

☆ agr / Ke bad (..) aye to pop Karo.

/.../~~a/..~~/b/~~c/..~~/d/./.

/.../b/d/./

→ (/.../b/d answer)

ignore Krna hai [.]

→ agr [..] aya to pop().

"/home/user/Document/../pictures",

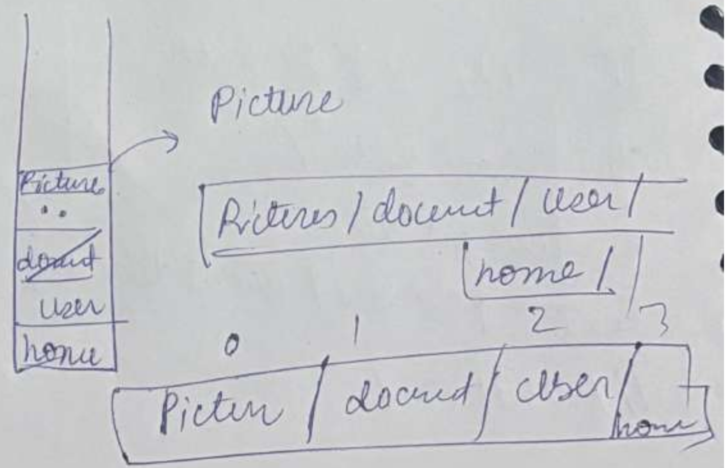
parts = ["home", "user", "Document", "..", " Pictures]

temp = "home", —

→ "user", —

→ "Document", —

→ .., —



```
① vector<string>s;
  string temp = "";
 for(int i=0; i<path.size(); i++)
  {
    if (i== part.size() || path[i]=='/')
      {
        if (temp=="" || temp==".") {
        }
        else if (temp == "..")
         { if (!st.empty()) st.pop-back();
         }
        else {
              st.push_back(temp);
         }
       temp.clear();
    }
    else {
          temp += path[i];
    }
  }
```

LC→ 402

Remove K digits

num= 143219   k = 3

→ remove 3 digits & make the number smaller

→ Since we have to maintain adjancey, use monotonic stack

if left > right remove it.

i.e. (1219) ans   (43,2) removed

Dry Run:

1, (4 > 3) X

1(3 > 2) x   k = 3

1(2 = 2) X

(1219) ✓

① add to stack

→ if (s.top >= cur)
  s.pop();

→ else push to stack;

② if (k>0 && !s.empty())
{
  s.pop(); // at top the largest element
}

③ make string, from stack

④ remove leading zero's

```
# for(int i=0; i< num.size(); i++)
  {
      int x = num[i]-'0'
      while(!st.empty() && x<=st.top() && k>0)
      {
          s.pop();
          k--;
      }
      s.push(x);
  }
→ while(!st.empty() && k>0) st.pop();
→ String ans = " "
  while(!st.empty())
  int a=s.top();
char b = a-'0'
  ans = ans+b;
  }
reverse(num1.begin(), num2.~~begin~~end());

//remove leading zero's.
 int i=0;
 while( num2[i]==0 && i<num.size()) i++;
~~return num(substr(i))~~;
num1 = num1.substr(i);
return num1;
```

Char to int

'0' → 48
'1' → 49
'2' → 50.

char ch = '7'

☆ [int x = ch - '0']

int to char ⇒ ☆ [char ch = x + '0']
