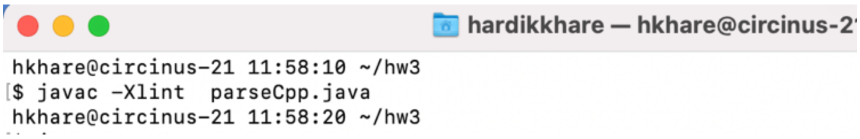
HARDIK KHARE | 70765344

HW 3.2 (50 points)

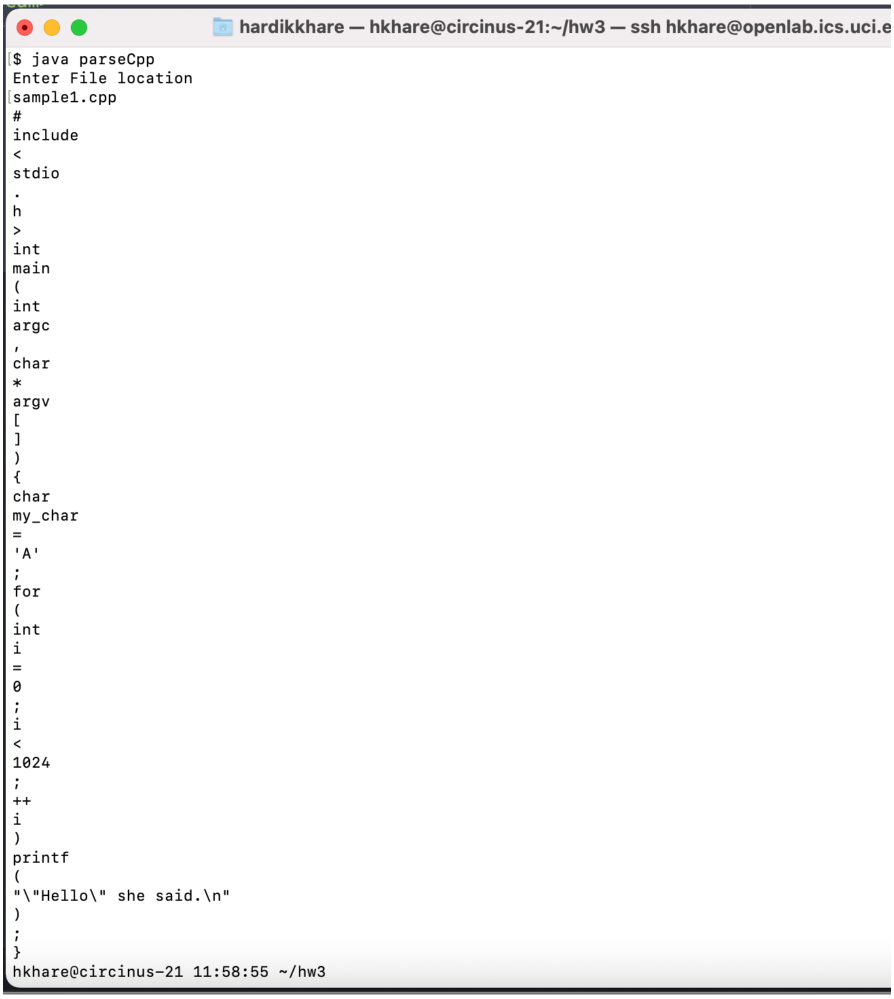
Write a program, called parseC, to parse a C++ program into tokens. You should recognize and discard program comments (/* to */ or after //).

Section 1: Successful compilation of program



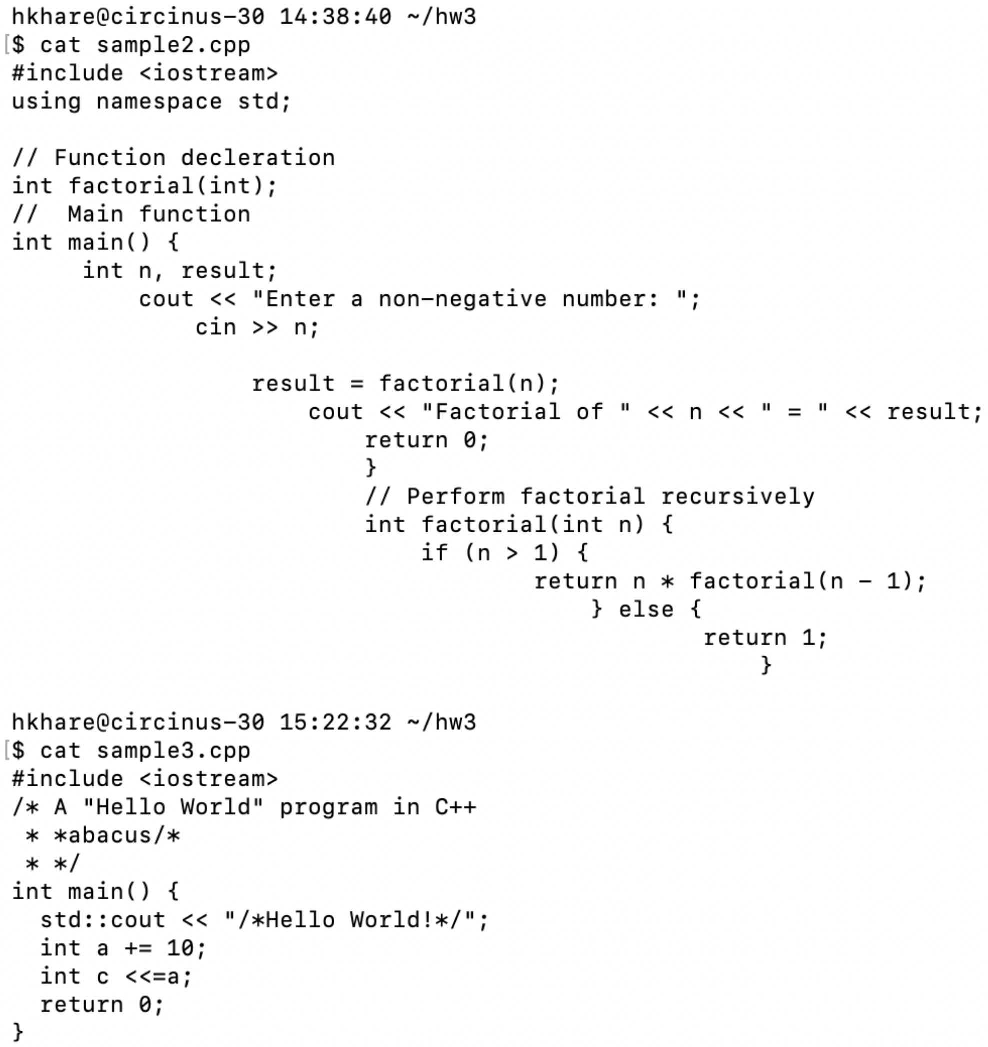
```
hkhare@circinus-21 11:58:10 ~/hw3
[$ javac -Xlint  parseCpp.java
hkhare@circinus-21 11:58:20 ~/hw3
```

Section 2: program running on the provided example from the assignment


hardikkhare — hkhare@circinus-21:~/hw3 — ssh hkhare@openlab.ics.uci.e

```
[$ java parseCpp
Enter File location
[sample1.cpp
#
include
<
stdio
.
h
>
int
main
(
int
argc
,
char
*
argv
[
]
)
{
char
my_char
=
'A'
;
for
(
int
i
=
0
;
i
<
1024
;
++
i
)
printf
(
"\"Hello\" she said.\n"
)
;
}
hkhare@circinus-21 11:58:55 ~/hw3
```

Section 3: Provided test input



```
hkhare@circinus-30 14:38:40 ~/hw3
[$ cat sample2.cpp
#include <iostream>
using namespace std;

// Function decleration
int factorial(int);
//  Main function
int main() {
     int n, result;
         cout << "Enter a non-negative number: ";
             cin >> n;

                 result = factorial(n);
                     cout << "Factorial of " << n << " = " << result;
                         return 0;
                         }
                         // Perform factorial recursively
                         int factorial(int n) {
                             if (n > 1) {
                                     return n * factorial(n - 1);
                                         } else {
                                                 return 1;
                                                     }

hkhare@circinus-30 15:22:32 ~/hw3
[$ cat sample3.cpp
#include <iostream>
/* A "Hello World" program in C++
 * *abacus/*
 * */
int main() {
  std::cout << "/*Hello World!*/";
  int a += 10;
  int c <<=a;
  return 0;
}
```

Section 4: Edge Case #1

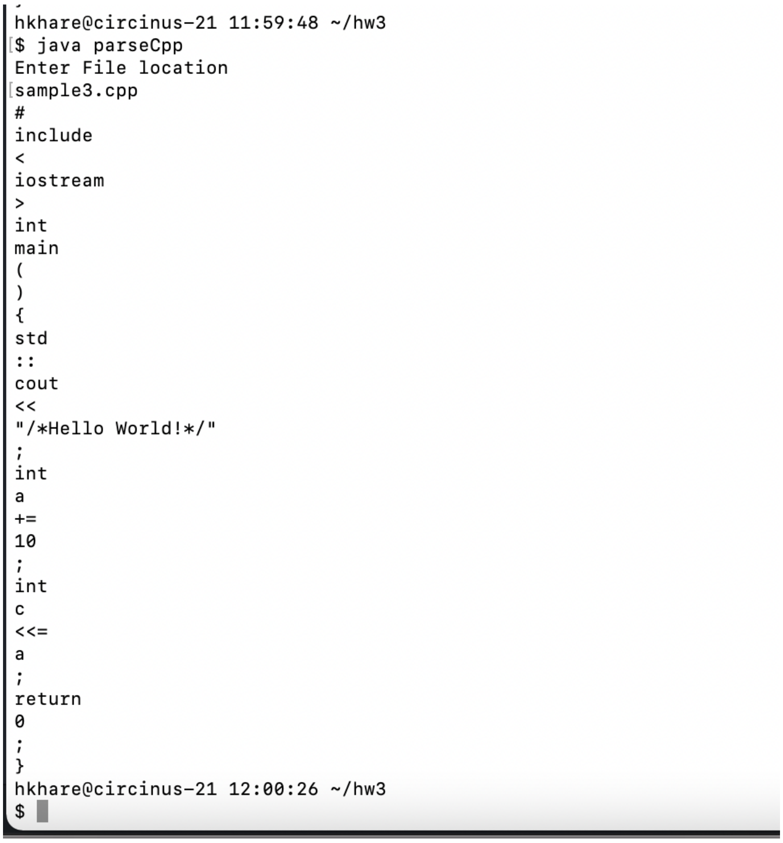
Description: Multi-line comments with many /* characters within the comment

Input:

```
#include <iostream>
/* A "Hello World" program in C++
*abacus/*
*/
int main() {
 std::cout << "/*Hello World!*/";
  int a += 10;
  int c <<=a;
  return 0;
}
```

Expected Output: Program is able to successfully decode correct closing multi-line comment.

Output:



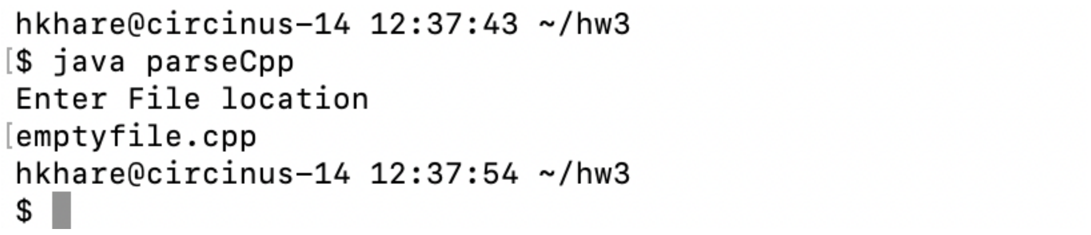
```
hkhare@circinus-21 11:59:48 ~/hw3
$ java parseCpp
Enter File location
sample3.cpp
#
include
<
iostream
>
int
main
(
)
{
std
::
cout
<<
"/*Hello World!*/"
;
int
a
+=
10
;
int
c
<<=
a
;
return
0
;
}
hkhare@circinus-21 12:00:26 ~/hw3
$
```

Section 5: Edge Case #2
Description: Empty file

Input: {empty file}

Expected Output: Program will not be able to tokenize anything and should print empty output.

Output



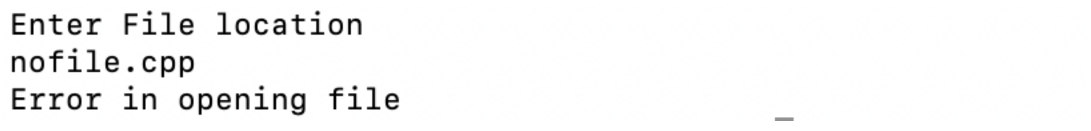
```
 hkhare@circinus-14 12:37:43 ~/hw3
[$ java parseCpp
 Enter File location
[emptyfile.cpp
 hkhare@circinus-14 12:37:54 ~/hw3
 $
```

Section 6: Edge Case #3
Description: input file does not exists

Input: non-existent cpp file

Expected Output: Program will not be able to open the file and should print error handling statement.

Output



```
Enter File location
nofile.cpp
Error in opening file
```

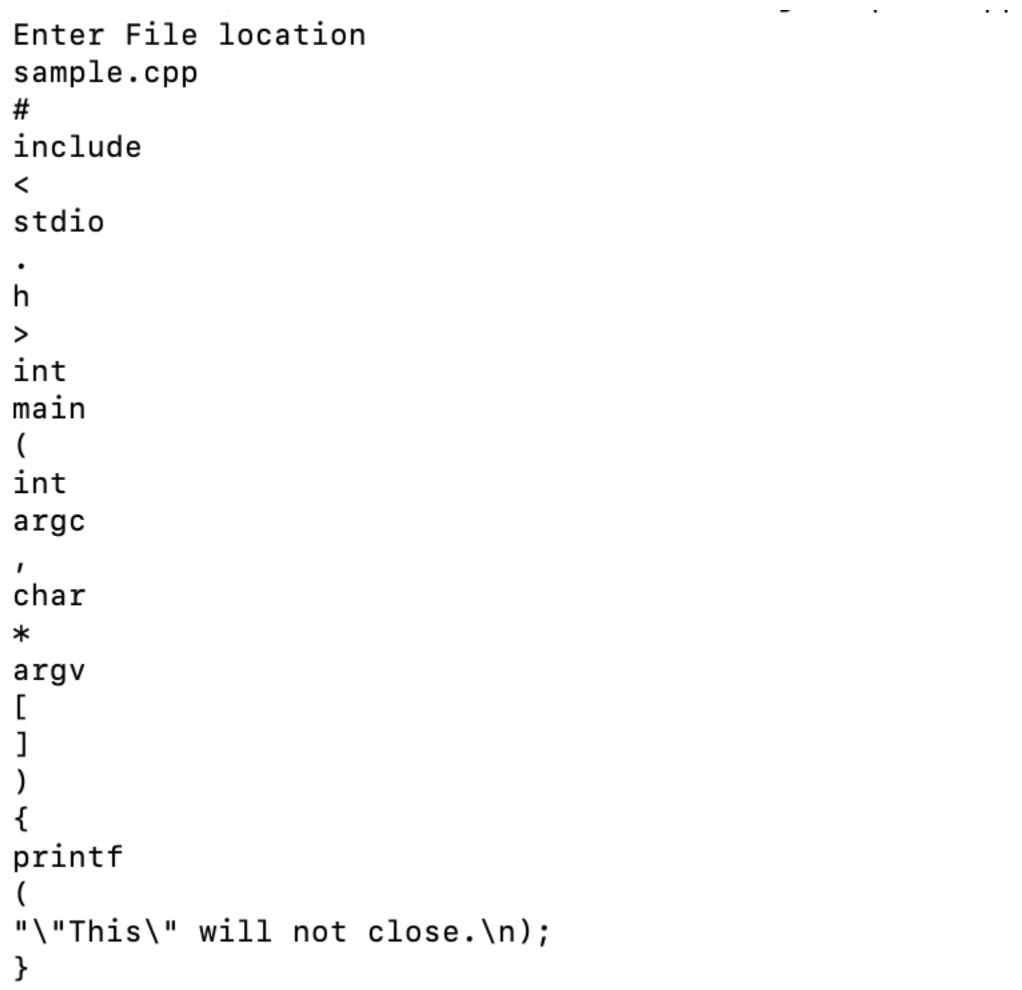
Section 7: Edge Case #4
Description: There is no closing quote.

Input:

```
#include <stdio.h>
int main(int argc, char *argv[])   /* here is a comment */
{
            printf("\"This\" will not close.\n);
}
```

Expected Output: Program will still print everything after last open quote

Output:

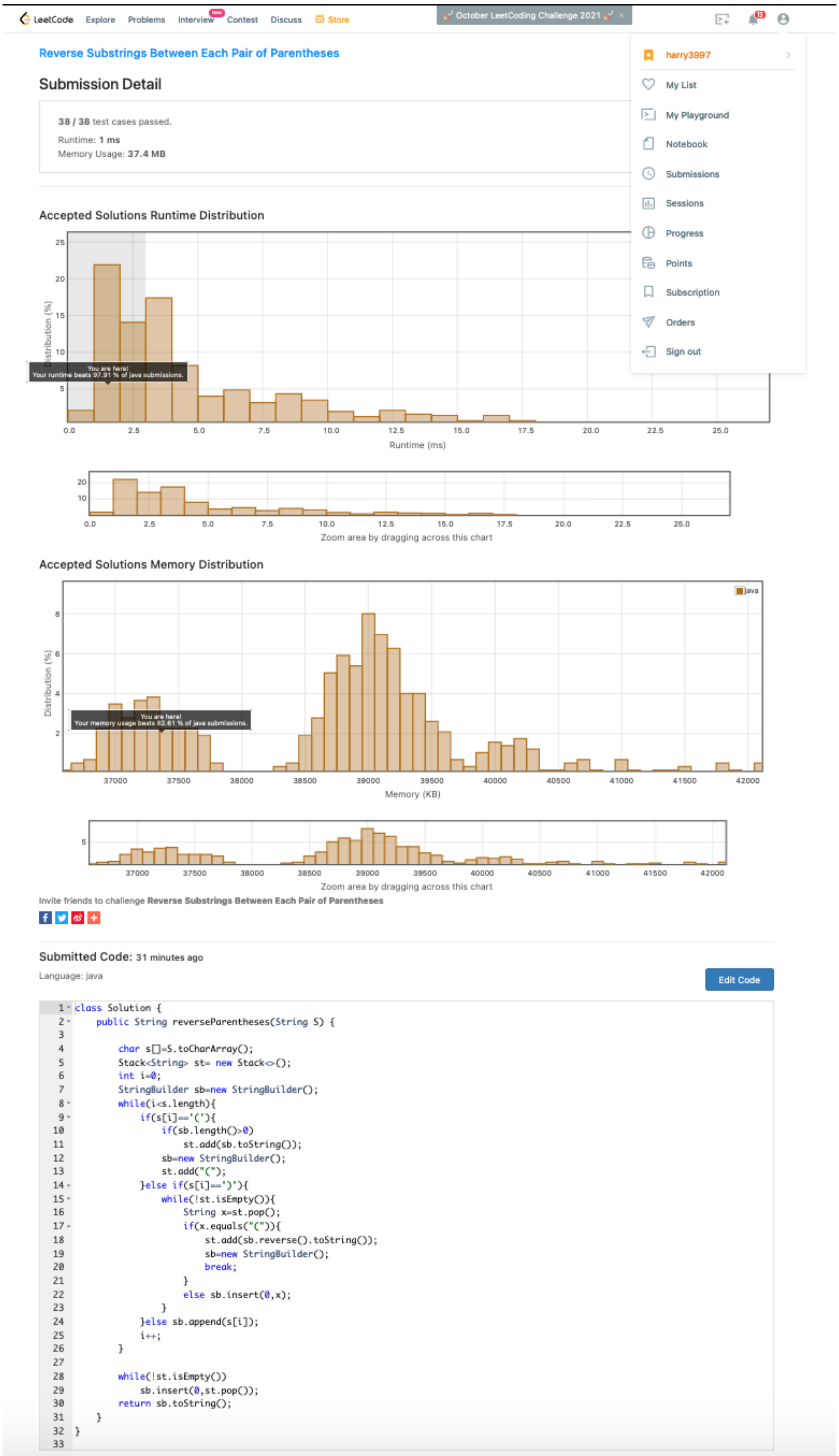


```
Enter File location
sample.cpp
#
include
<
stdio
.
h
>
int
main
(
int
argc
,
char
*
argv
[
]
)
{
printf
(
"\"This\" will not close.\n);
}
```

Note: above, last '}' is printed as part of string after opening quote

Leetcode problems 1190. Reverse Substrings Between Each Pair of Parentheses and 946. Validate Stack Sequences given below.

https://leetcode.com/problems/reverse-substrings-between-each-pair-of-parentheses/



Reverse Substrings Between Each Pair of Parentheses

Submission Detail

38 / 38 test cases passed.

Runtime: **1 ms**

Memory Usage: **37.4 MB**

Accepted Solutions Runtime Distribution

You are here!
Your runtime beats 97.91 % of java submissions.

Accepted Solutions Memory Distribution

You are here!
Your memory usage beats 82.61 % of java submissions.

Invite friends to challenge **Reverse Substrings Between Each Pair of Parentheses**

Submitted Code: 31 minutes ago

Language: java

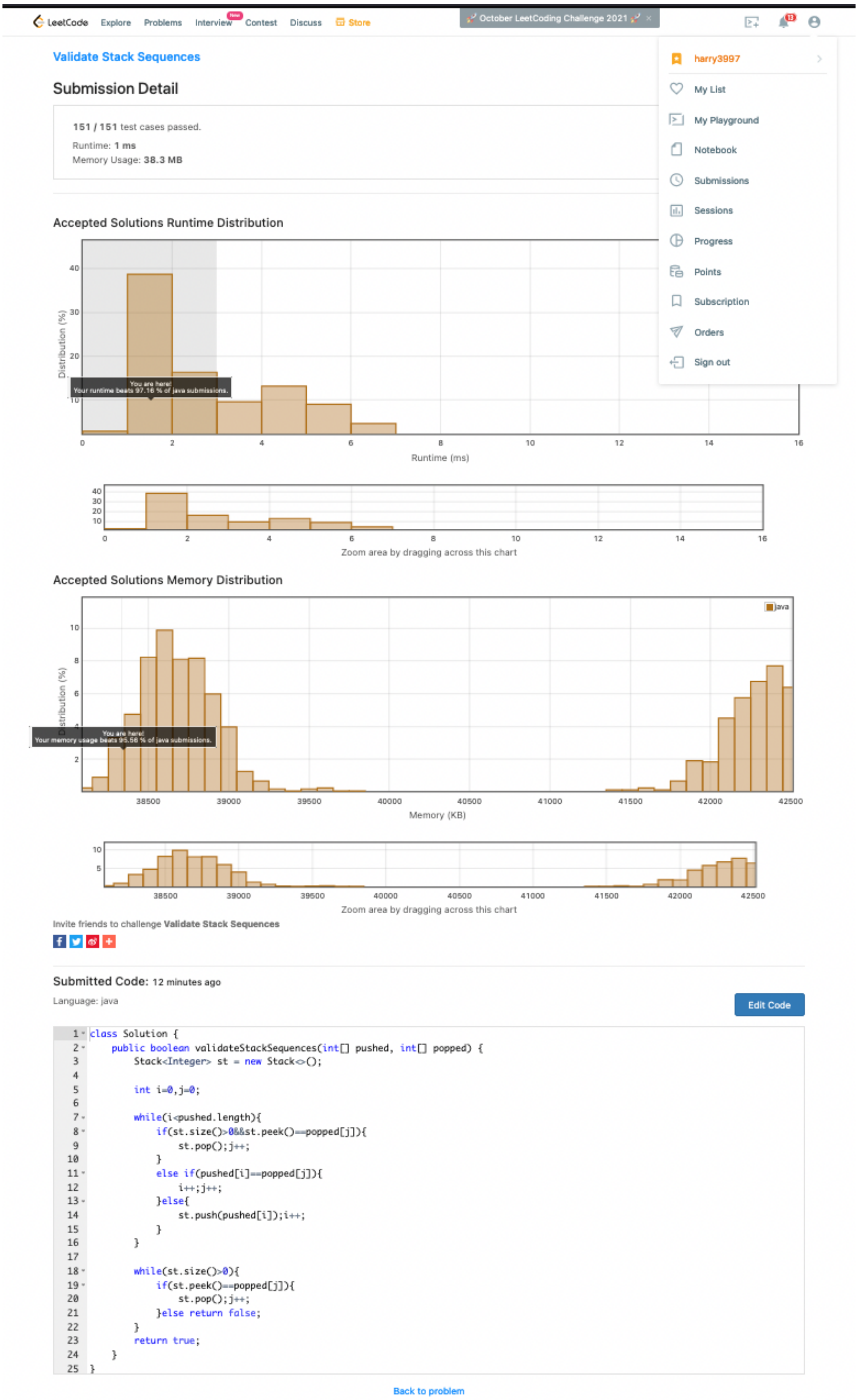
```java
class Solution {
    public String reverseParentheses(String S) {

        char s[]=S.toCharArray();
        Stack<String> st= new Stack<>();
        int i=0;
        StringBuilder sb=new StringBuilder();
        while(i<s.length){
            if(s[i]=='('){
                if(sb.length()>0)
                    st.add(sb.toString());
                sb=new StringBuilder();
                st.add("(");
            }else if(s[i]==')'){
                while(!st.isEmpty()){
                    String x=st.pop();
                    if(x.equals("(")){
                        st.add(sb.reverse().toString());
                        sb=new StringBuilder();
                        break;
                    }
                    else sb.insert(0,x);
                }
            }else sb.append(s[i]);
            i++;
        }

        while(!st.isEmpty())
            sb.insert(0,st.pop());
        return sb.toString();
    }
}

```

https://leetcode.com/problems/validate-stack-sequences/



Validate Stack Sequences

Submission Detail

151 / 151 test cases passed.

Runtime: 1 ms

Memory Usage: 38.3 MB

Accepted Solutions Runtime Distribution

You are here!
Your runtime beats 97.16 % of java submissions.

Accepted Solutions Memory Distribution

You are here!
Your memory usage beats 95.56 % of java submissions.

Invite friends to challenge **Validate Stack Sequences**

Submitted Code: 12 minutes ago

Language: java

```java
class Solution {
    public boolean validateStackSequences(int[] pushed, int[] popped) {
        Stack<Integer> st = new Stack<>();

        int i=0,j=0;

        while(i<pushed.length){
            if(st.size()>0&&st.peek()==popped[j]){
                st.pop();j++;
            }
            else if(pushed[i]==popped[j]){
                i++;j++;
            }else{
                st.push(pushed[i]);i++;
            }
        }

        while(st.size()>0){
            if(st.peek()==popped[j]){
                st.pop();j++;
            }else return false;
        }
        return true;
    }
}
```