Ryan Thompson
Assignment 3
Data Structures
10/14/2022

Assignment 3

Data Structures

Stacks

Submission Notes

This assignment requires two file submissions including one cpp file for question 1 (which is a programming assignment) and one PDF or JPG file for question 2 (which is not a programming assignment). Do not upload a code snippet for question 1, the only files you submit for the code should be the the source file. Do not submit any zip or compressed files, binary files, or any other generated files. Do not put the code inside the PDF file. Submission of unnecessary files or binary files in addition to source files will make you lose the points of the assignment. The code must have the following:
a. Use consistent indentation through your code.
b. Use consistent braces style through your code.
c. File-level comments, these comments will include the file name, your name, course number, and date of last modification.
d. Function level comments include the function's description, the function's parameters, and the function's return value. These comments will appear before each of the functions, not the prototypes.
e. In function comments, these comments will include a description of the atomic functionalities within the bodies of the functions.
f. We evaluate your code using the following C++ online compiler. You will not get points if your assignment does not compile with this.

`https://www.onlinegdb.com/online_c++_compiler`

Question 1 (70 points)

Using the **array** implementation of stacks implement a program which gets an input expression and checks whether the parenthesis used in that expression are balanced or not.

Example:
Input(cin): $[A * \{B + (C + D)\}]$
Output(cout): Balanced

Input(cin): [{()()}]
Output(cout): Balanced

Question1.cpp uploaded to canvas assignment

Input(cin): $[A * \{B + (C + D\})]$
Output(cout): Not Balanced

Input(cin): [{(])()}
Output(cout): Not Balanced

Your program should have the following components:

```
class Stack {
public:
   int top;
   char exp[100];
   Stack() { top = -1; }
   // functions prototypes
   bool push(char item);
   char pop();
};

int isMatchingPair(char character1, char character2);

int areParenthesisBalanced(char exp[]);

int main() {
  char exp[100];
  cout<<"enter the expression:\n";
  cin>>exp;
  if (areParenthesisBalanced(exp))
    cout<<"Balanced";
  else
    cout<<"Not Balanced";
  return 0;
}
```

Hint:
Combine and modify *Stacks_ArrayImplementation.cpp* and *BalancedPar.cpp* codes from week 5 and 6 of class.

Question 2 (30 points)

Convert the following in-fix expression into its corresponding post-fix expression using stack operations. Show the algorithms by filling out the following chart.

$$10 + (3 * 5 + 1) * 3 - 8/2$$

step	symbol	stack	Postfix
1	10		10
2	+	+	10
3	(	+ (	10
4	3	+ (	10 3
5	*	+ (*	10 3
6	5	+ (*	10 3 5
7	+	+ (+	10 3 5 *
8	1	+ (+	10 3 5 * 1
9	)	+	10 3 5 * 1 +
10	*	+ *	10 3 5 * 1 +
11	3	+ *	10 3 5 * 1 + 3
12	-	-	10 3 5 * 1 + 3 * +
13	8	-	10 3 5 * 1 + 3 * + 8
14	/	- /	10 3 5 * 1 + 3 * + 8
15	2	- /	10 3 5 * 1 + 3 * + 8 2
			10 3 5 * 1 + 3 * + 8 2 / -