CO1212 Practical work on CO1222 (2019/2020)

EUSL/TC/IS/2019/COM/24

ICA-02

1. On top, we can see three declarations and initializations of some variables.
    1. [Line 3]: There is an integer variable named ***top***, that's been initialized to -1.
    2. [Line 4]: And there's another integer variable called ***limit***, that's been initialized to 1000.
    3. [Line 5]: And the third, there's an array of characters (**char**) named ***myStack*** and it is initialized with a new instance of a character array with the size of the value of ***limit*** – in this case 1000. So essentially, it is a character array that has the size of 1000.

2. [Line 71]: First, in the ***main()*** method – there is an expression written and assigned to a String variable called ***expr***.
3. [Line 72]: Then the length of that String is assigned into an integer (int) variable called ***len***, by calling the ***length()*** method on the String. In this code, the length of the ***expr*** string is 16. So, ***len*** integer is initialized to 16.
4. [Line 74]: Then there's an if statement that calls the ***isBalanced()*** method with parameters ***expr*** and ***len***.
5. [Line 7]: That call to the ***isBalanced()*** should be evaluated before proceeding to the next line. So, the ***isBalanced()*** method is executed with ***expr*** and ***len***.

    6. [Line 9]: In there, first – there is an if condition that checks if the expression string is null **or** if length of that string is odd (using modulo operation on ***len***).
    [Line 11]: If either of that are true, the ***return*** keyword will be executed with the return value ***false***.
    But in this case, since ***expr="{t[(a+b)*(c+d)]}"*** (not null) and ***len=16*** (not odd), both of them are false, so the execution continues from line 14 (because line 13 is a blank line).

    7. [Line 14]: Then there is a for loop. It iterates from integer variable ***i***'s value 0 – to the value of ***len*** (which is the size of the expression string).
    In this program, ***len = 16***. So, this loop will be executed for 16 times, if break, continue, return keywords will not get executed.

        8. [Line 16]: Inside the loop, first we assignes the ***i*** th character of the expression string to a char (character) variable named ***c***.

        9. [Line 17]: Then there is an if condition that checks whether that character is some kind of an opening bracket *(-[-{* .

        10. [Line 19]: If that evaluates to true, then the ***push()*** method is called with the parameter of ***c***.
        So, it will be called with the ***i*** th character of the expression string.

        11. [Line 40]: The ***push()*** method gets executed with the given character value to the variable ***p*** and,
            1. [Line 42]: Simply increase the value of the integer variable ***top*** by 1, by calling the unary increment (++) operator.

2. [Line 43]: Then the value of ***p*** is assigned to the ***myStack*** array's element position of the value of ***top***.
3. [Line 44]: And the execution returns back to the caller.

12. [Line 22]: Then there is another if condition that checks whether the character at ***i*** th index of the expression string is some kind of a closing bracket)-]-} .
    And, if that evaluates to true,

    1. [Line 24]: First, it calls the ***isEmpty()*** method.

       [Line 61]: Inside the ***isEmpty()*** method, it checks whether the value of the integer variable ***top*** is equal to -1. [Line 63]

       This is used to check whether no character has yet been added to the ***myStack*** array, because the value of ***top*** would be incremented otherwise.

       [Line 64]: If that evaluates to true, the method ***isEmpty()*** returns true.
       [Line 66]: Otherwise, it returns false to the caller.
       This method ***isEmpty()*** is used to check if the array ***myStack*** is still empty.

       [Line 26]: If the ***isEmpty()*** method returns true, which means – if ***myStack*** is in fact empty, the ***return*** statement gets executed with the value ***false***.

       If that is not true, the return statement is skipped, and the execution falls to line 29.

    2. [Line 29]: Then we initialize a ***local*** character variable called ***top*** with the value returned by executing the ***pop()*** method.
       i. [Line 48]: Inside the ***pop()*** method, we first initialized a character variable called **c** by taking the element at the index of ***top***'s (the integer member variable) value and assigning it to **c**.
       ii. [Line 49]: Then the integer member variable ***top*** is decremented by one, by executing the unary decrement operator.
       iii. [Line 50]: Then finally, the character **c** is returned to the caller.

    3. [Line 31]: Then there is an if condition.
       It checks whether that character variable ***top*** is an opening bracket of some kind, **and** the current value of character **p** is **not** a closing bracket **of the same kind**.
       [Line 33]: If that evaluates to true, the **return** statement is executed with the value **false**.

Or if the check to see if the character is a closing bracket evaluates to false, the execution skips all these above steps and falls into line 35.
----- The ***for-loop*** execution finishes an iteration. -----

13. When the for loop is finished executing by finishing all iterations or by returning in the middle, the execution goes to line 36.
    [Line 37]: There is a call to ***isEmpty()*** method (which checks whether the ***myStack*** array is empty), and it returns the value that was returned by the ***isEmpty()*** method to the caller which is ***main().***

14. In this code, with the expression ***"{t[(a+b)*(c+d)]}"***, the ***isBalanced()*** method finishes it's loop and all other checks without returning false in the middle.

    It iterates ***i*** through all 16 characters and checks till the array ***myStack*** is in fact empty, then finally returns the value true by the returned value of ***isEmpty().*** [Line 37]

15. [Line 74]: The returned value ***true*** (in this case) makes the if condition execute line 75.
16. [Line 75]: Since ***isBalanced()*** method returned the value true, line 75 is executed.
    That prints out to the console ***"This expression is balanced ".***

17. [Line 76]: Since the if statement evaluated to true, the else block skips execution.
18. [Line 78]: The ***main()*** method finishes execution, and the program exits.