Assignment No. - 10

Sumit Gulab Bhamare
SEComp A08
Sub - DSL

Aim: To illustrate the various concept of stack.

Problem Statement: In any language program mostly syntax error occurs due to unbalancing delimiter such as (), {}, []. Write C++ program using stack to check whether given expression is well parenthesized or not.

Learning Objectives: To understand concept of stack
To analyze the working of various functions of stack

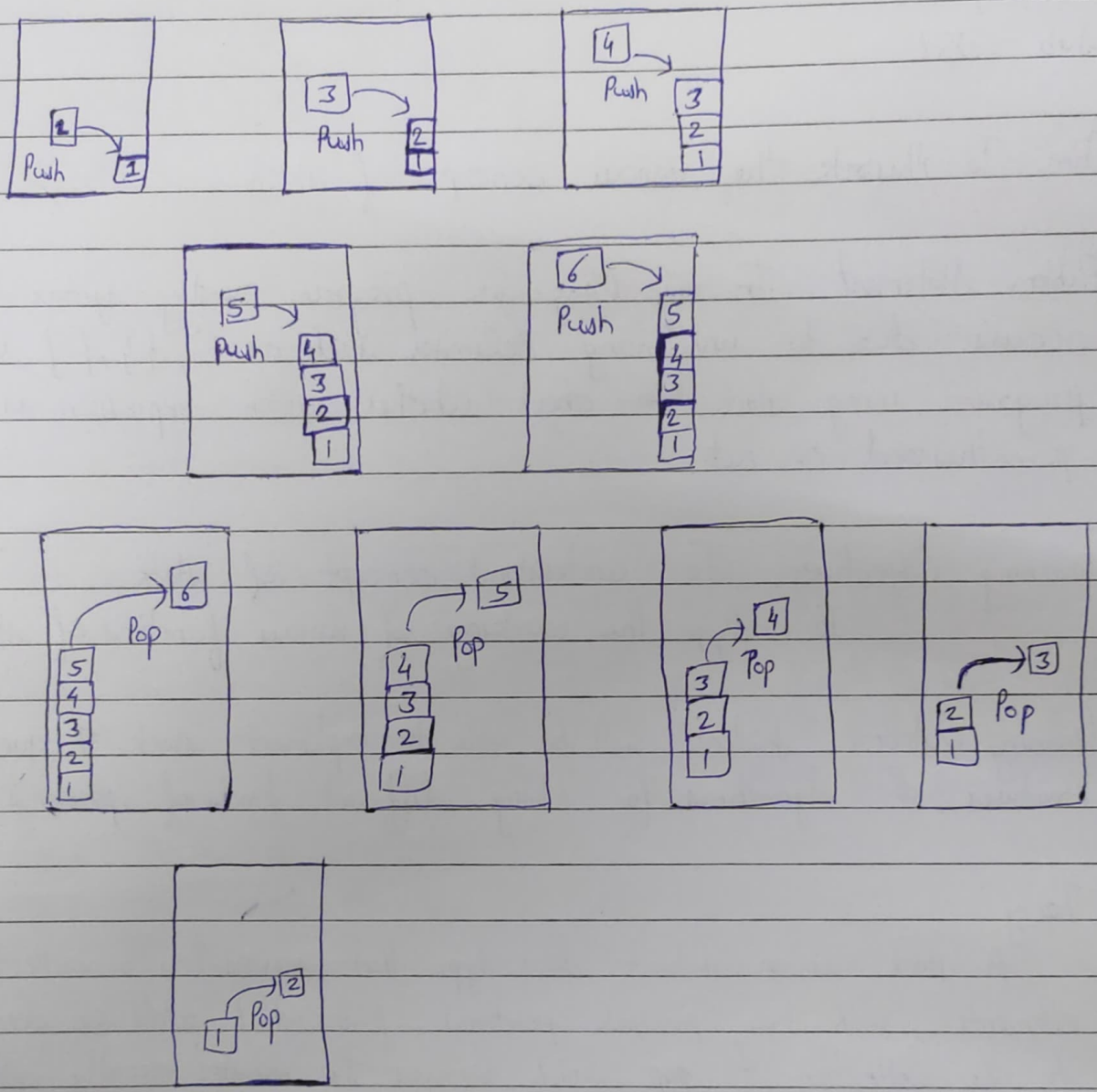
Learning Outcome: Students will be able to implement stack & queue data structures & algorithms for solving different kinds of problems.

Theory:

A stack is an abstract data type that serves as a collection of elements, with two principal operations: push, which adds an element to the collection, & pop, which removes the most recently added element that was not yet removed. The order in which elements come off a stack gives rise to its alternative name, LIFO (for last in, first out). Additionally a peek operation may give access to the top without modifying the stack.

The name "stack" for this type of structure comes from the analogy to a set of physical items stacked on top of each other, which makes it easy to take an item off the top of the stack.

while getting to an item deeper in the stack may require taking off multiple ~~order~~ other items first.



Balanced parentheses:-

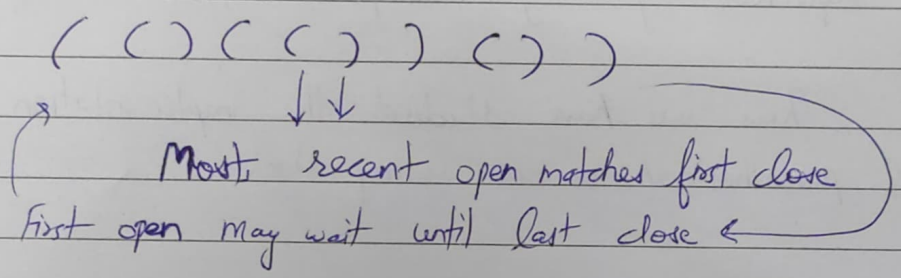
We now turn our attention to using stacks to solve real computer science problems. You have no doubt written arithmetic expressions such as

$$(5+6)*(7+8)(4+3)(5+6)*(7+8)(4+3)$$

where parentheses are used to order the performance of operations

Balanced parentheses means that each opening symbol has a corresponding closing symbol & the pairs of parentheses are properly nested.

The ability to differentiate between parentheses that are correctly balanced & those that are unbalanced is an important part of recognizing many programming language structures.



Input : Enter any expression having numbes of opening & closing brackets.

Output : Entered expression is well parenthesized or not.

Algorithm :

Step 1: Declare a character stack S

Step 2: Now traverse the expression string exp.

If the current character is a starting bracket ('(' or '{' or '[') then

push it to stack

If the current characters is a closing bracket (')' or '}' or ']') then

pop from stack &

if the popped character is the matching starting bracket then

fine

else parenthesis are not balanced

Step 3: After complete traversal, if there is some starting bracket left in stack then "not balanced".

Step 4: Exit.

Software required: g++ / gcc compiler - / 64 bit fedora.

Conclusion: Thus we have studied the implementation of expression is well parenthesized or not using stack.