TED UNIVERSITY

CMPE 223-01 Fall 2021

Programming Homework 1

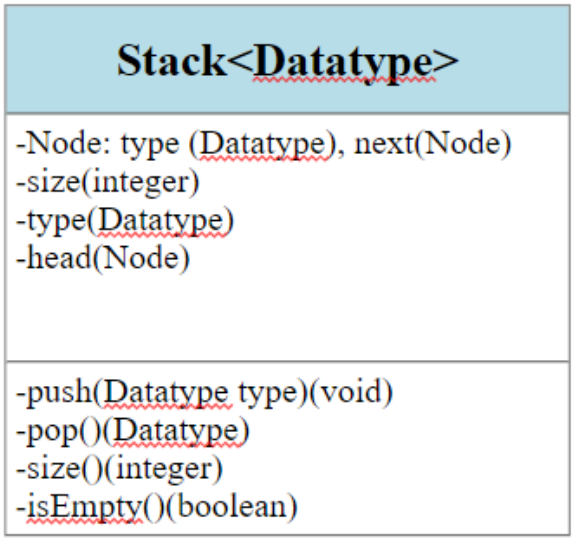
Name Surname: Yakupguly Malikov

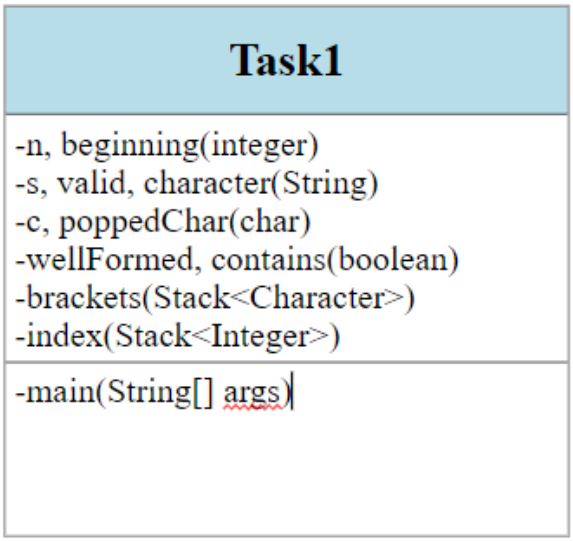
Section: 01

Problem Statement and Code Design

Task 1

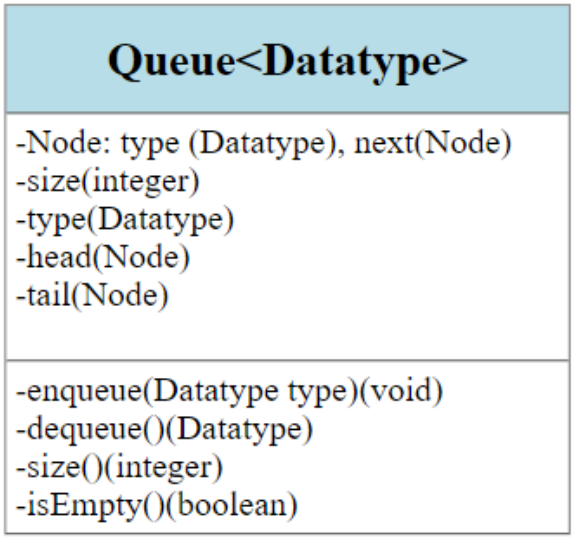
First, I get inputs from Tester1 class and put the expressions to string array and number of arithmetic expressions to an integer variable n. After, the program checks if the given expression is valid. If it is not valid then the program prints that it is not valid and if it is valid then it works on the expression. When there is opening bracket, it pushes it to brackets<Character> and it pushes index to another index<Integer> (I use index to print the expression between two brackets and starting, ending position). If there is closing bracket, the program controls whether there is opening bracket in stack if there is not any then it means that the given expression is not in well-formed, so it prints it and stops to calculate. If there is opening bracket in stack, it checks if two brackets match. If they do not match then the program breaks and if they match it prints the starting, ending indexes and the expression between two brackets.

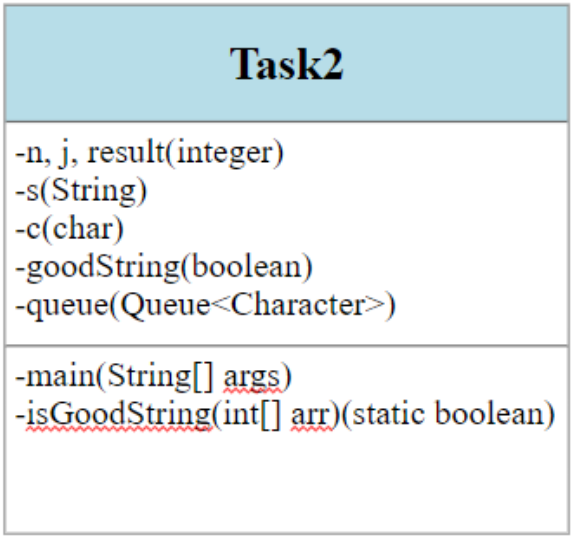




Task 2

In task2, like in task1 I get input values from Tester2 class and put the strings to array of strings and the number of given strings to an integer variable n. First, it checks the given arr is good string using the isGoodString method (after removing the previous i's character if it is still good string then the program should do not have to enter while loop). In while loop, the program each character then increases the index of that character from arr (use this to check if it is good string or not by counting each character) and enqueues it to queue. After checking if the string is good string, it quits the while loop because there no need to check other (because it is already good string, adding any character to a good string makes it good string). After while loop, if the program finds good string, then it removes the first character from queue and array continues to search other good strings.





Implementation and Functionality

Task 1

First, I took integer from class. After, used loop in order to get each expression n times and removed whitespace. Initialized two stacks, one is character stack and other is integer stack. Character stack is for storing brackets and integer stack is for storing indexes in order to write substring and check if the string is well formed. wellFormed is for checking the string is well formed and contains is for if the string contains other than operators, bracket types and integer. In the for loop in line 26 is to check if the string contains any other characters that is not in valid. If string does not contain any other character, it calculates expression and if string contains it prints invalid parse. In the for loop in line 38, program iterates in each character and if there is opening bracket it pushes bracket to brackets stack and pushes index to index stack. If there is closing bracket, it controls if brackets stack is empty or not. If brackets stack is empty, then it means that string is not well formed. If stack is not empty it calculates whether the string is well formed or not by controlling the popped opening bracket matches with the closing bracket. After, it prints the indexes of substring and the substring. In the end, it prints if the string is well formed or not.

In the stack class, I created Node class with the values type and next. Also, I used push(Datatype type), pop(), isEmpty(), and size() methods. Push method puts the given value to stack, pop method takes out the last pushed value from stack, isEmpty method controls if the stack is empty or not and the size method gives the size of stack. Other than this, initialized head and size variables.

Task 2

I get an integer from Tester2 class that determines how many strings user wants to enter. After, get the string in order to calculate how many good string pairs are in the string. In the for loop, I controlled if the given characters form the good string. If not, then enters the while loop and takes a character from string. It increases the index of each character in arr and enqueues it to queue. After, it controls if there is any character left to from good string. If isGoodString method return true, then while loop stops because we find good string. After while loop, if goodString is true, then it removes the first character from good string and increases result. In the end, prints the result. In the isGoodString method, controlled if there is any character left to form the good string. It returns false if there at least one or more than one 0 in the array because there is still one or more than one character(s) left to form good string.

In the queue class, I created Node class which has type and next. There is head, tail, size variables and enqueue(Datatype type), dequeue(), isEmpty() and size() methods. Enqueue() method puts the given value in the parameter to queue, dequeue() method removes the first enqueued value from queue, isEmpty() checks whether the queue is empty or not and size() method returns size of the queue.

Testing

Task 1

In Tester1 class, I provided 3 inputs and method to get them. getNumberOfExpressions() method return the number of expression to evaluate and getExpressions() method return array which is expressions to evaluate.

Task 2

In Tester2 class, likewise in tester1 class, there is 3 inputs and getNumberOfStrings() method returns the number of strings to evaluate and getStrings() method return the array of strings to evaluate.

Final assessment

One of the trouble points in completing this assignment was in second question which finding a way or logic to solve the task and applying it to queue. The most challenging part was the task 2 because it was hard to understand to the question and find logic to solve it. I mostly liked task2’s being challenging, and it makes us to learn a lot in one question. I learned linked-list based stack and queue.