جامعة الإمام عبد الرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY
كلية العلوم والدراسات الإنسانية - علوم الحاسب
College of Science and Humanities - Computer Science

Project Case Study
Stack

Prepared by

Students names :	ID:
Fatimah Hussain AL-hmood	2200004124
Wadha Nayef Alsheddi	2200003246
Asma Zaher Alshehri	2200000484
Albandari Saleh Alsanhani	2200006257
Dana Rami Kurdi	2200005678

Prepared for

Dr. Enas El-Sharawy

T.A Hessa Almutari

Date:7 /4 / 20222

Executive Summary

We developed a calculator that has an order of operations using stack method that receives the input from the user and extracts the operands and operators, performs calculation by using stack to arrange the operations, and finally gives out the results on the screen. The order that we will use to simplify expressions is called the order of operations, by following this order, we can solve all the problems using PEMDAS which stands for:

P – Parentheses
E - Exponents
M- Multiplication
D- Division
A - Addition
S - Subtraction

The Case Study

We have faced an issue with the normal simple calculator, because it doesn't prioritize the operations so we needed to find an alternative, therefore, we developed a program that will help to solve this problem by using the stack structure and we choose it because it used to solve such problems like expression evaluation

Challenges

The main challenge that we face is that the calculator used by us calculates the numbers without prioritize the ordering operations, which leads to errors in the outputs.

Solutions

-isOperator

Boolean method it return true when the user input one of the right operators (+ , - , * , /)

-getprecedence

This method in our code is to gives the operators their correct order in the High Priority

-processOperator

First, we defined 2 variables for OP1 and OP2.

in the if condition we check if the stack is empty by using method isEmpty() , if true display("Expression error") , and change the Boolean variable error to true . But in the else block we assign the data which is in the top in variable (a)(these 2 blocks is for OP1) and we repeat it with different variable for OP2 . Then we defined variable r for result , it chose the correct operator and compute it by using if else if .

Then change error Boolean variable to true again . Finally push the result variable to the stack

-processInput

We created an array of type String to take the expression, we pass the tokens inside a loop and test it if the character if is a number between 0 and 9, we push the value into the value stack but if the character is an operator first we will test it if the (operator stack) is empty Or if the priority of the character is greater than the priority of the character at the top of the operator stack) we will push the character inside the operand stack, but if (the character is not empty and the priority of the character is less than the character at the top) we pop for the top in the operator stack and pass it in processOperator() Then we push the character, but if the character is '(' then we push it and if ')' we will pop the character at the top and pass it inside the processOperator() as long as the operator stack is not empty and The character at the top isOperator(), but If the stack is not empty and the character at the top of the operatorStack is ' (' we will pop the operatorStack , if it does not meet one of the conditions then there will be an error (unbalanced parenthesis) then we will empty out the operator stack at the end of input and print the result if no error has been seen.

-Main

In the main method we will ask the user to insert the expression so the program can calculate and than show the results to the user . Also We created an object from class FullCalculater to call the method processInput and perform the correct operation on it.

-<stack<charcter

stack<double>

Since we didn't use nor identify posh() , pop() and isEmpty() we choose to use these Because they are more efficient and they have all the methods related to Stack. Also note that we will use the keyword "Peak" instead of "top "in this method. Lastly it showcase our extracurricular knowledge .

Results

We decided to use Stack structure because it is easy to implement and allow memory control that way we can save memory and only use what we need. Also its easy to add and remove from it and has fasted time access.

Conclusion

In conclusion by building this project it showed us the importance of stack structures in solving our daily problems and how we can implement it to such problems and solve them easily and fast. also it showed us the importance of system devolvement process that provides good solutions to problem solving