# **Stacks Report**

# Design

## Introduction

For this assignment it would have been very difficult to make a calculator that could convert equations from the conventional way that we as humans write them for a stack to understand. This presented a problem as a Stack could not process the equation when it is written out in what is known as Infix notation. This problem could be fixed if calculator instead used Reverse Polish notation or RPN where the operators follow their operands. This means that instead of entering “2 + 2” the user would have to enter “2 2 +” for instance.

## Requirements

***Functional:***

*R1. The system should allow the user to add multiple integers up.*

*R2. The system should allow the user to subtract multiple integers from each other.*

*R3. The system should allow the user to multiply several integer.*

*R4. The system should allow the user to add, subtract and multiply integers in the same equation.*

*R5. The system should display an error when an invalid character is inputted.*

***Non-Functional:***

*R6. The system should be user friendly.*

*R7. The system should properly inform the user how to use a delimiter in the equation.*

*R8. The systems menu should be easy to use.*

I was able to tackle and successfully complete all of these requirements however I would have liked to have had division as a requirement.

## Pseudocode

### inputCheck method

*Prints out “Please enter an equation – eg 1 2 +”*

*Prints out “Enter 0 to exit”*

*Creates a Scanner*

*Assigns the line of scanner to a new String called “input”*

*If the input is 0*

*Return null*

*Creates a String array called “sliceData” and sets it to split on a space*

*Returns sliceData*

### calculate method

*Creates an object of the Stack Class*

*Creates a foreachLoop for String called “element” which sliceData is assigned to*

*Declares “var” integer and sets it to the parsed integer*

*Push method is called for var*

*Catches if there is an exception*

*Declares an integer “a” and sets it to the called pop method*

*Declares an integer “b” and sets it to the called pop method*

*Declares an integer “result” and sets it to the operatorCalc method*

*Calls the push method and passes result*

*Catches if there is an exception*

*breaks*

*if the length of the int in stack is not 1 character*

*throw a new exception displaying “Wrong input”*

*Declares a new integer “result” and sets it to the called pop method*

*Prints the answer*

*Catches the exception and prints out message*

*Returns nothing*

### operatorCalc Method

*Switches operator to case if “-“ is entered*

*Returns a-b*

*Switches operator to case if “+“ is entered*

*Returns a+b*

*Switches operator to case if “\*“ is entered*

*Returns a\*b*

*Switches to default operator if one of the specified cases is not met*

*Throws a new “Unknown operator” exception*

|  |
| --- |
| **RPNException** |
| * error: String - exception |

|  |
| --- |
| **List** |
| * Listnode |
| * addToList() * deleteFromStart() |

## Class diagram

Creates object

Creates object

Creates object

Uses exception

Uses exception

Creates object

|  |
| --- |
| **RPNCalculator** |
| * rpnStack: Stack |
| * menu() * processEquation() * inputCheck() * calculate() * operatorCalc() * rpnInformation() |

|  |
| --- |
| **Stack** |
| * List * length: int |
| * push() * pop() |

|  |
| --- |
| **ListNode** |
| * integer: int * next: ListNode |
| * getNext() * getInteger() * setNext() |

|  |
| --- |
| **Main** |
| * calc: RPNCalculator |
| * main() |

Creates object

|  |
| --- |
| **Tester** |
| * stackInstance: Stack * testCalc: RPNCalculator |
| * stackTest() |

Creates object

## AC12001 – Test Plan

Name: …Isaac Lowry…………………………………………………………………………..

Matric number: …170025555 …………………………………………………….……….

Lab Title: …AC12001 assignment 1: Stacks………………………………..……….

Test number/date/version: 28/01/18 ……………………………………….……..

Test Notes: …Tests run on submitted assignment ……………………………

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Description** | **Test Data** | **Expected result** | **Worked?** |
| Popping from a non-empty stack | 10 | 10 is displayed | Y |
| Popping from an empty stack | N/A | Outputs empty stack message | Y |
| RPN Calculation | 2 3 + | 5 | Y |
| RPN Calculation | 2 3 4 + | Outputs “Wrong input” error | Y |
| RPN Calculation | 2 + | Outputs “Wrong input” error | Y |
| RPN Calculation | 5 3 - | -2 | Y |
| RPN Calculation | 3 5 - | 2 | Y |
| RPN Calculation | 4 4 \* | 16 | Y |
| RPN Calculation | \* | Outputs “Wrong input” error | Y |
| RPN Calculation | 0 0 \* | 0 | Y |
| RPN Calculation | a b + | Outputs “Wrong input” error | Y |
| RPN Calculation | 1 + 2 | Outputs “Wrong input” error | Y |
| RPN Calculation | + 5 5 6 \* | Outputs “Wrong input” error | Y |
| RPN Calculation | - 6 - | Outputs “Wrong input” error | Y |
| RPN Calculation | 5 h \* | Outputs “Wrong input” error | Y |

# Self-evaluation

I found this assignment relatively challenging as I had to learn the concept of both stacks and also the Reverse Polish Notation scheme. Once understood I also found it difficult to convert the String from the scanner into integers. However, I was able to overcome all of this thanks to research on the internet. I found the calculation to also be challenging but in the end, through similar research I was able to complete it. I found this to be the most rewarding part of the assignment as the purpose of it had been met. I was quite happy with my use of custom exception handling which I learnt about from research as it proved to be very useful. Overall, I feel my effort was sufficient and I am proud of the assignment however I found myself unable to complete all of the optional extras due to my limited coding experience and also somewhat poor time keeping. I feel if I was to attempt this again I would give myself more time to research and work on the assignment, so I could include extras.