Software Requirements Specification

for

Complex Calculator

University of Gloucestershire Introduction to Programming Fundamentals

Prepared by Michael Thomas

s1605827

05/02/2017

Table of Contents

1. Introduction

1.1 Purpose

The goal of this project is to provide a scientific calculator for users needing to solve expressions of any length.

1.2 Document Conventions

The format of this SRS is simple. Bold face and bold titles for general topics or specific points of interest. The rest of this document will be written using the standard font arial.

1.3 Intended Audience

The intended audience is mostly students who are just starting college who need a scientific calculator but cannot afford one but have seen or used one before.

1.4 Product Scope

Refer to CT4021_Michael_Thomas_s1605827_Calculator_Project.docx

1.5 References

This SRS was modeled to the one found online at
http://moodle.glos.ac.uk/moodle/course/view.php?id=10983
The original creator is named at the bottom of each page

2. Overall Description

2.1 Product Perspective

The calculator is a new piece of software designed and created by myself. It will provide the means to solve expressions such as "(2(2+3)*(5/2))^3" but they can be more complicated.*

2.2 Product Functions

These are the major functions to enter and solve any expression;
Input Expression
Press "="
Answer is outputted

2.3 User Classes and Characteristics

The calculator will be used by the frequent user group, people who are going to use it for a few hours a day, referring back to the intended audience students do tend to spend hours revising and studying in class using a calculator. Also being a student the user group would of been expected to of seen a scientific calculator before, and thus knowing the layout meaning for a easier and quicker understanding to how to use this one.

2.4 Operating Environment

Because this is a C++ program it can run on Linux, Windows, Mac, Free BSD and most other POSIX-compliant OSes. It also supports i386, PPC, IA32, IA64, Arm (and some other) architectures

2.5 Design and Implementation Constraints

There are no constraints at this point in time.

2.6 User Documentation

A readme file will be put with the product to help the user understand how to use the product.

2.7 Assumptions and Dependencies

There will be no need for extra documentation beyond this SRS for the user to utilize this product.

3. External Interface Requirements

3.1 User Interfaces

There will be only one interface, the interface which holds the calculator. This is where the user can input the desired expressions and receive the answers.

4. System Features

4.1 Solving Expressions

4.1.1 Description and Priority

This is the base of the entire project, it needs to convert the inputted expression (being infix notation) to reverse polish notation (formally known as postfix notation) using the shunting yard algorithm so the calculator can use a RPN solver to solve the

expression and output the answer.

4.1.2 Stimulus/Response Sequences

User will input an expression and press the "=" button.

4.1.3 Functional Requirements

REQ-1: Expression is in infix notation

REQ-2: Expression has no errors in it

If the expression is not valid an error will be returned

REQ-3: User pressed "="

5. Other Nonfunctional Requirements

5.1 Software Quality Attributes

Usability, the GUI will be made so that the user will grasp the controls quickly through the use of affordance, as the keypad layout is similar to a physical calculator and the one built in most operating systems.

Reusability, the Tokenizing method used in the code makes adding more operations, unary operators and much more very easy for the person adding the change due to the Tokens being a class holding as much information as the programmer desires.

Maintainability, code should be neat and well commented so if a programmer needs to come and look at the code to fix/change/add something anyone picking it up will be able to understand what does what.

Reliability, the calculator will always return the correct result as the methods used to convert and calculate are highly maintained and coherent as well as it doing its job correctly.

Interoperability, due to the user group being students they would of already seen many calculators so due to the accordance of it we can implement almost anything in this calculator and the user will already know what it means and does.