Project 1: SRPN

Atul Anand

January 5, 2017

Abstract

This is a project that you have to write in the programming language C. You can use any Integrated Development Environment (IDE) for the development of your scripts, but **your scripts have to run on the Linux shell without requiring the installation of libraries, modules or other programs**.

This project is due on the 19$^{\text{th}}$ of January 2017.

1 Learning Objectives

At the end of this project you will be able to design and write a medium-sized program using the appropriate procedural software techniques of data encapsulation and decomposition.

2 SRPN

Whilst performing some maintenance on a legacy system you find that the system uses a program called **SRPN**. **SRPN**, unfortunately, is not documented and you are not able to obtain the source code and no one seems to know who wrote it. So your boss tells you to rewrite in C.

Your task is to write a program which matches the functionality of **SRPN** as closely as possible. *Not that this includes not adding or enhancing existing features.* **SRPN** is a reverse Polish notation calculator with the extra feature that all arithmetic is saturated, i.e. when it reaches the maximum value that can be stored in a variable, it stays at the maximum rather than wrapping around.

You will be given the working model of **SRPN** that you found during doing some maintenance of the legacy system. To run the program, execute:

```
java -jar SRPN.jar
```

2.1 Basic arithmetic support

Program must be able to input at least two numbers and perform one operation correctly and output the following.
Input:

```
10
2
```

```
+
=
```

Output:

```
12
```

Input:

```
11
3
-
=
```

Output:

```
8
```

Input:

```
7
7
*
=
```

Output:

```
49
```

Input:

```
12
4
/
=
```

Output:

```
3
```

Input:

```
10
2
%
=
```

Output:

```
0
```

Input:

```
10 2 * =
```

Output:

```
20
```

2.2 Multiple operators and operand support

Program must be able to handle multiple numbers and multiple operations.
Input:

```
3
3
*
4
4
*
+
=
```

Input:

```
1234 2345 3456 d + d + d =
```

2.3 Saturation handling

Program must be able to correctly handle saturation.
Input:

```
2147483647
1
+
=
```

Input:

```
-2147483647
1
-
=
20
-
=
```

Input:

```
100000 0 - d * =
```

2.4 Miscellaneous features

Program must include the less obvious features of SRPN. These include, *but are not limited to...*:
Input:

```
1
+
```

Input:

```
10
5
-5
+
/
```

Input:

```
11+1+1+d
```

Input:

```
# This is a comment
1 2 + # And so is this
d
```

Input:

```
3 3 ^ 3 ^ 3 ^=
```

Input:

```
r r r r r r r r r r r r r r r r r r r r r d r r r d
```

3 Assessment

3.1 Marking

Key issues for marking the code will be:

- compiling
- running with expected input
- robustness (handling incorrect user input)
- module design
- proper use of data encapsulation and decomposition
- algorithm being used

You stand to lose marks for repeated, badly-structured or documented code. Marks will also be deducted for ill-named variables or functions or for inconsistent indentation. In cases where where it is not clear from the assignment how it should be marked, you may be asked to explain or demonstrate your program.

Your program will be tested on the following inputs and others that are similar. Successfully completing each step will give you 15 marks each. The remaining marks 40 marks are for program structure and comments, for a total of 100 marks. Your pass mark is 75%.

4 Submission Instructions

The deadline for this project is **5pm 19th January 2017**.

Your program solutions should be a ZIP file which should be named `SRPN-yourname.zip`. The ZIP file should be extractable and the directory should contain the files necessary to compile and run your code `SRPN.c` with any additional class files needed to allow your program to compile and run.

Before you submit your solution, make sure that the ZIP file contains all necessary files and creates the correct directory.