Assignment 1: Fix it – **In, Post**

Dr. William Kreahling

February 26, 2021

1 Overview

For this assignment you will write a program in Rust that reads postfix notation mathematical expressions from a file and solves them. The program will then print out the expressions re-written in infix notation, along with their solutions. The solutions will be printed out in with the expressesion with the smallest answer coming first.

2 Details

The following sections will describe each file that you will need, and what should appear in each of those files.

Your program will accept two command line arguments (AKA Program Arguments). The first is the name of the input file, the second in the name of the output file. If either argument is missing, then print a usage message and quit.

2.1 `Expression`

You will create a class (or the Rust equivalent) called *Expression* that contains three fields.

1. postfix: A String that holds a single postfix expression
2. expr: A vector of f64's used to calculate the numerical solution
3. infix: A vector of Strings, used to calculate the infix version of the expression

Expression will have the following:

1. A constructor that accepts one line of input (from the file)
2. A method called solve which solved the postix expression and creates the equivalent infix expression. This method does not have a return value.

2.2 `Functions`

You will also have the following function:
references should be mutable when appropriate

1. build_expression_list: This function accepts a reference to a string slice representing the input file name and returns a 'Result' with a vector of expressions from the file or an appropriate error.
2. solve_list. Takes a reference to a vector of Expressions and solves them. No return value.
3. sort_list. Takes a reference to a vector of Expressions and sorts them based on the value of the expressions solution.
4. write_to_file: This takes a reference to a string slice, representing the output file name and a reference to a vector of expressions. This function returns a 'Result'
5. main: Parses and handles command line argument, and contains the logic and code to run the program. If there is an error writing to the output file, main should print an appropriate error message.

You may use the following crates. You may not use any other crates without written permission of the instructor.

```
use std::env::args;
use std::fs::File;
use std::io::{BufRead, BufReader, Error, Write};
use std::process::exit;
```

3 Sample Run

Below is a *simple* example of the program in action.

3.1 ex.dat

```
1 2 +      3     4    - *


  2 6 +
```

3.2 out.dat

```
( 1 + 2 ) * ( 3 - 4 ) = -3
2 + 6 = 8
```

3.3 Program Execution.dat

```
> cargo run
    Finished dev [unoptimized + debuginfo] target(s) in 0.06s
     Running `target/debug/rpn`

Usage: `cargo run [input file] [output file]`

> cargo run --release
   Compiling rpn v0.1.0 (/home/wck/classes/common/cs370/os_rust/class_projects/rpn)
    Finished release [optimized] target(s) in 1.38s
     Running `target/release/rpn`

Usage: `cargo run [input file] [output file]`

> cargo run --release ex.dat
    Finished release [optimized] target(s) in 0.00s
     Running `target/release/rpn ex.dat`

Usage: `cargo run [input file] [output file]`

> cargo run --release ex.dat out.dat
    Finished release [optimized] target(s) in 0.01s
     Running `target/release/rpn ex.dat out.dat`
```

4 Style

I expect that *every* line of code to be properly formatted; for this assignment we'll just the standard Google guidelines. If your code is not formatted correctly, your grade will suffer.

I expect your program to be appropriately commented. Note that Rust is not Java; JavaDoc doesn't apply. If you are missing expected documentation, your grade will suffer.

Don't let your grade suffer; do it right, take pride in your work! ☺

5 Notes on Collaboration

You must work in teams of 2 on this assignment, but I will not assign groups. Note that *in general* all members of a team will receive the same grade on the assignment – choose your teammates wisely! If, however, I am convinced by your partner(s) that you didn't pull your weight, you grade will suffer (again, take pride in your work). If you don't want to do the assignment in a group, let me know and I'll let you be in a group by yourself.

6 Hand-In Instructions

This assignment is due by 11:59 PM on MOnday, March 15. A **single version** version of the assignment is due from each team. Submit all source files associated with the program as well as the `Cargo.toml`. To submit your files, use the *handin* command on agora. Handin works as follows:

```
handin.<course#>.<section#>  <assignment#>  <files>
```

Submit the entire directory with the *Cargo.toml* file as well as the *src* directory.

```
handin.370.1  1  prog1.tbz
```

You may use tar with zip, 7zip or bzip2 to archive/compress your files.