CS-102 Project 3 – Palindromes

Professor: Brian Frost

Summer 2019

Specifications

A palindrome is a word or phrase which reads backwards the same as it does forwards, such as racecar or noon. Palindromes can also be phrases, such as:

- Madam, I'm Adam.
- A man, a plan, a canal: Panama!
- Able was I, ere I saw Elba.

Your task is to write a program which reads strings line-for-line from standard input, and outputs PALINDROME or NOT A PALINDROME depending on whether or not the given string is a palindrome. Strings will contain at most 50 characters, so the input string should be stored in a buffer of that size. Whitespace, case and special characters should be ignored, or else many palindromes won't register as palindromes.

As an example, if my input is a file that looks like

```
Racecar
I am trapped in this text file
hoobleelbooh
WHATAMI!I~~~MATA hw
```

My output should be

```
PALINDROME
NOT A PALINDROME
PALINDROME
```

You will create 2 strings – a forward and reverse string – which you must compare to one another. Your restriction imposed by your evil instructor is that you must store the reverse string in as little memory as possible, freeing up the used memory after each line has been checked. Good style here (checking for available memory, redirecting the pointer after freeing) is important for your grade!

I suggest you maintain a top-down design approach, using functions to perform most tasks. This will not affect correctness, but will make your code much more readable, and thus will affect your style score.

Submission and Grading

Please submit your assignment as a .c file to b.frost@columbia.edu by midnight on Wednesday, July 31. Remember that lateness is penalized! Please state which operating system, text editor and compiler you used in your email. Also, please don't hesitate to ask as many questions as you need through email, or by coming to office hours.

Grading is broken down according to **Correctness** and **Style**. Your total grade is out of 100 points.

Correctness (80 points): This is simply how well your program fits the exact above-stated specifications.

Style (20 points): This is composed of efficient commenting, intuitive spacing and indentation, intuitive variable names, and overall elegant code. Unnecessary statements, poor placement of variable instantiations and otherwise difficult-to-read code will result in a loss of points here. Remeber to put a brief comment at the top of your code explaining the program's operation.