CS352 Assignment #4
Three Programming Problems

Due Date: Wednesday, April 1 at 7:00 PM

Turnin:

Submit all files to the `cs352s15assg4` turnin directory. Recall that this command looks like

`turnin cs352s15assg4 <list of files>`

You may list the files individually, or, if they are the only .c files in your directory, you may just use the ‘*.c’ as the second argument.

You may check what files have been submitted by typing "turnin -ls cs352s15assg4".

Your scripts will be put through a series of test cases and compared to ours using diff. This means that things like spelling, punctuation and whitespace are very important if you would like to pass all the test cases without needing to appeal. Additionally, your filenames will need to match exactly in order for the script to pick it up. If you have any questions about the output of your scripts or the format of the filenames, please stop by our office hours, post on Piazza or email us.

Remember, no late work will be accepted.

The following rules apply to all programs:

1. **Do not use anything covered after chapter 13 in the King textbook.**
2. Each program should be a single C file.
3. Do not #include anything beyond `<string.h>`, `<stdio.h>` and `<stdlib.h>`
1. **prefix arithmetic (prefix.c)**

Please write a program that will evaluate a prefix notation arithmetic expression on integer values.

1. The program will read in a prefix arithmetic expression and evaluate it.
2. Prefix expressions are composed of integer literals and the operators, +, -, *, /, and whitespace.
3. All integer literals will be non-negative
4. All integer values may kept in “int” variables.
5. All values and operators are separated from each other by whitespace.
6. You may not make any assumptions about the complexity of the expression.
7. Operators have C semantics.
   a. E.g., “+ 3 4” is an expression that would evaluate to 7.
   b. “* + 5 4 3” would print 27.
8. If there is extra non-whitespace input after the expression, the program should not print anything and exit with status 1. Extra trailing whitespace is ok.
9. Exit with status 1 for division by zero.
10. If there are no input expressions, then exit with status 1.
11. If there is an error in the input, the program should exit silently with status 1.
12. If there were no errors, the program should exit with status 0 after printing the computed value on a single line followed by a newline.

2. **Multi-Precision Multiplication (multiply.c)**

Please write a program that will multiply two large integers.

1. Your program will read these two potentially huge non-negative integers from user input.
2. The numbers will be separated by a newline.
3. The program will then multiply these two numbers and print out their product, followed by a newline. There will be no additional spacing before/after the sum.
4. The numbers will be of the form digits with no decimal point.
5. Legal input will have no more than 100 digits for each number. Output, therefore, could be up to 200 digits.
6. Exit with status 1 for illegal input. Illegal input would be anything that has extra or illegal characters.
7. Your answers must not lose any digits--these values will not fit in C’s given types, so you will need to construct them out of arrays.
8. After printing the value, exit with status 0.
3. **Find (find.c)**

Please write a program that will look for words in input strings.

1. Your program will read input from stdin. It will process one line at a time.
2. The program should read only the first 40 chars of each line and ignore any additional chars on each line.
3. The program will take command line arguments, which represent the word(s) to be searched for.
4. The program should print every line of input that contains at least one occurrence of at least one of the command line arguments. No line should be printed more than once.
5. The program should exit with status 0.
6. If there are no command line arguments, then nothing would be printed.