McGill University
School of Computer Science

COMP-206

Mini Assignment #4

Due: March 25, 2022 on myCourses at 23:55

Do the following for this assignment:

- This is an individual assignment. You need to solve these questions on your own.
- You MUST use `mimi.cs.mcgill.ca` to create the solution to this assignment. An important objective of the course is to make students practice working completely on a remote system. Therefore, you **must not** use your Mac command-line, Windows command-line, nor a Linux distro installed locally on your laptop. You can access `mimi.cs.mcgill.ca` from your personal computer using `ssh` or `putty` as seen in class and in Lab A. If we find evidence that you have been instead using your laptop, etc., to do some parts of your assignment work, you might lose all the assignment points. Your solutions must be composed of commands that are executable in `mimi.cs.mcgill.ca`.
- A testing script, written in Bash, has been provided with this assignment. Please test your program with this script. **Note**: the TA will add additional tests to the script. It is suggested that you do so as well when preparing your solution.
- No points are given for commands not covered in class or that do not work.
- The assignment is graded proportionally by the TA. The TA will not modify your solution in any way to make it work.
- Please read through the entire assignment before you start working on it. You can lose up to 3 pints for not following the instructions in addition to the points lost per question.
- Lab F provides some background help for this mini assignment.
- Total points: 20.

Question 1: Divisible (5 points in total)

Write a C program that prompts the user for 3 integer numbers and then verifies if the numbers are all divisible by the first number without a remainder AND that the three number are in increasing order. The program returns an error code of 0 if the numbers are divisible by the first number and in increasing order, it return an error code of 1 if the numbers are not divisible but in increasing order, 2 if they are divisible but not increasing, or an error code of 3 if the numbers are not divisible and not increasing. In addition to returning those error codes it will also print to the screen: "`Divisible & Increasing`", or "`Not divisible & Increasing`", "`Divisible & Not increasing`", or "`No divisible & Not increasing`". The program terminates after doing this one time.

Example Execution:

```
Please input three numbers: 10 5 7
Not divisible & Not increasing

Please input three numbers: 3 6 9
Divisible & Increasing

Please input three numbers: 5 20 10
Divisible & Not increasing
```

Do the following:

- Write your program in the filename `divisible.c`
- Compile using **gcc** and the executable program must be called `divisible`.
- The program must run at the command line as follows: `./divisible`
- Your program must work with the provided testing script: `mini4q1tester.sh`
- Add your own tests to the script to fully test your program.

What to hand in:

- File: `divisible.c`

Question 2: Anagram (15 points in total)

According to Wikipedia, an anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once. For example, the word **anagram** can be rearranged into **nagaram**, or the word **binary** into **brainy**.

Create a C program called `anagram.c` (compiled as the program `anagram`) that implements the anagram check on 2 words entered from the command line. Specifically, the user will run the program by typing the following at the command line prompt: `./anagram WORD1 WORD2`. Note: you are not checking if the anagram is a proper word. You will only check whether all the letters appear in WORD2 from WORD1. The program requires the user to input 2 words then returns the error code 0 if the words are anagrams and print to the screen "`Anagram`", or the error code 1 otherwise and print to the screen "`Not an anagram`". The program does this once and then terminates.

Example Execution:

```
./anagram mary ramy
Anagram

./anagram bob lary
Not an anagram
```

Do the following:

- Write your program in the filename `anagram.c`
- Compile using gcc and the executable program must be called `anagram`.
- The program must run at the command line as follows:
  `./anagram WORD1 WORD2`
- Your program must work with the provided testing script:
  `mini4q2tester.sh`
- Add your own tests to the script to fully test your program

What to hand in:

- File: `anagram.c`

WHAT TO HAND IN

Everything must be submitted to My Courses before the due date. Remember that you can hand in your assignment up to two days late but there will be a penalty of 5% each day. After that, your assignment will not be accepted. Please hand in the following:

- divisible.c
- anagram.c

HOW IT WILL BE GRADED

The assignment is worth a total of 20 points.

Grades Deducted:

- -3 prints for not following instructions
- Late day points

Grades Awarded:

- Question 1 – 5 points
  - +1 reading the input
  - +2 divisible test
  - +2 increasing test
- Question 2 – 15 points
  - 2 points for getting the 2 words as inputs
  - 5 points for returning the proper exit codes
  - 8 points for the anagram algorithm

GRADING RULES

The following rules are followed by the TA when grading assignments:

- A program must run in order to get a grade (even if it does not run well). If it does not run (does not compile) it will receive a zero. (Make sure to run your programs from Trottier – they sometimes do not run the same from home when logging in using putty.)
- The TA will grade using the mimi.cs.mcgill.ca server.
- All questions are graded proportionally (assuming the program runs at all). This means that if 40% of the question is correct, you will receive 40% of the grade.
- The TA will compile the c code then runs the submitted script and observe the execution.
- The TA may modify the script to provide other words for checking.
- The TA will finally check whether the script displays the messages properly depending on the anagram check result.