CS 161 Assignment #1 – The Fortune Teller
Due: Sunday, 1/12/2020, 11:59 p.m.

Goals:

- **Write, compile, and run your own program**
- **Gain familiarity with the standard input/output library (`iostream`) to interact with users**
- **Declare, assign, and use appropriately typed variables to store information**
- **Learn common compiler error messages and how to fix them**

Part 0. Sign Up for Your Demonstration

Every assignment in this course is graded by demonstrating your work for 10 minutes with a TA. You are required to **meet with a TA within two weeks after the due date** to demo. Please sign up for your slot now (yes, before you begin the assignment!) to ensure your work will be graded. You can schedule a demo with a TA on the course website (http://classes.engr.oregonstate.edu/eecs/winter2020/cs161-020) in the far right column of the bottom table labeled "Grading Hours". You can select one of your lab TAs, or a different TA. Be sure to specify

- **Tip:** Pick an early slot before they are all taken! The earlier your demo, the sooner you get feedback, which will help you do even better on assignment 2. ☺
- **Demo Outside 2 Weeks:** Assignments that are demonstrated later than the acceptable time period will be receive a 50 point deduction. (Also, we cannot guarantee a late grading slot.)
- **Demo Late Assignments:** Late assignments must still be demonstrated within the two-week demo period beginning from the assignment's original due date.
- **Missing a Demo:** If you miss your scheduled demo with a TA, you will receive a 10 point (one letter grade) deduction to that assignment for each demo missed.
- **No demo**: If you do not demonstrate your assignment, you will receive a 0.
- **Submissions that do not compile on the ENGR servers will receive a 0 for the implementation part**. Please test your code before submitting!

Your goal in the demonstration is to show your TA how to use your program, explain how it works, and answer questions. Be proud of your work and ready to show it off!

Part 1. Create a Fortune Teller

(10 pts) Write a secret fortune that predicts the user's future (good or bad) and includes **at least 3 sentences** with **at least 5 numbers.** Remove the numbers to create blanks.

- Example: "You will sell _(number)_ copies of your award-winning novel."
- Example: "Your _(number)_ children will battle over who inherits your _(number)_ stuffed alligators."
- Example: "You will discover a sunken pirate ship worth $_(number)_."

Be creative! To earn 10 points, you must use sentences other than these examples.

Create a Word or text file with **your name, assignment number (1), and the due date**. Put your secret fortune in a section called “Part 1”.

(60 pts) Write a C++ program that prompts the user to enter enough numbers to fill in the blanks in the fortune, then prints out the filled-in fortune.
Example:

- Computer prints: `Please enter a negative whole number:`
- User types: `-32`
- [Computer asks for 4 more numbers]
- Computer prints: `You will live in a town where the average winter temperature is -32 F.`
- [Computer prints sentences to use the remaining 4 numbers]

Download this .cpp file as a starting point:
http://classes.engr.oregonstate.edu/eecs/winter2020/cs161-020/assignments/assign1_fortune.cpp

Tip: Start with a fortune that has only one number to fill in, get that working, then add more numbers.

(A) (10 pts) Given our discussion of what each data type can store, decide on an appropriate data type to store each user input. (You will need at least 5 variables.) Try to use the smallest data type needed for each value.

- Example: I would use a `short` for the temperature example above, which gives me a range of (integer) values from -32768 to 32767.
- Overall, you must use at least **3 different types** from 8 options: `short`, `unsigned short`, `int`, `unsigned int`, `long`, `unsigned long`, `float`, `double`. You can modify your fortune template as needed to enable you to explore and use 3+ data types.
- Hint: Refer to **Rao Lesson 3 (pp. 39-46)**.

(B) (10 pts) **Prompt the user** to enter enough numbers to fill in the blanks, without knowing what the fortune says.

- Another example: “`Please enter a number with two decimal places:` ”
- Note: For this assignment, you do not need to check if the user followed your instructions correctly. We will learn about checking user input later!

(C) (10 pts) **Print out the fortune with the (5 or more) blanks filled in**.

(D) (5 pts) Use `std::cin` and `std::cout` to communicate with the user.

- Tip: You can use the std namespace to avoid writing “`std::`” each time.

(E) (5 pts) Use **informative names** for your variables (e.g., `number_of_children`, `lottery_value`, `body_temperature`). The user won’t see your code before running the program, so this won’t give anything away!

(F) (20 pts) Justify your choice of data type for each variable **in a comment above its declaration**. You must include the **minimum and maximum values** your chosen data type can hold.

- Examples:
```
// I assume number of children is between 0 and 100,
// so a char (min 0, max 255) is sufficient.
char number_of_children;

// I assume weight change (pounds gained or lost)
// can be positive or negative, but no fractions of a pound,
// so I am using a signed short integer (min -32768, max 32767).
short weight_change;
```

Here is an example of using the standard I/O library:

```
#include <iostream>

using namespace std; /* so we don't have to write std::cout */

int main() {
   uint age;          /* not initialized; user will fill in */
   cout << "How old are you? ";        /* no endl here; why? */
   cin  >> age;                        /* no endl here; why? */
   cout << "You are " << age << " years old." << endl;

   return 0;
}
```

(10 pts) **Program Style/Comments**

In your implementation, be sure to include a program header in your program (see example below). Use proper indentation and spacing. Include comments for someone reading your program to understand how it works. Be sure to review the style guidelines for this class and begin trying to follow them. For example, don't align everything on the left or put everything on one line!
http://classes.engr.oregonstate.edu/eecs/winter2020/cs161-020/assignments/cs161-style-guidelines.pdf

```
/******************************************************
** Program: fortune_teller.cpp
** Author: Your Name
** Date: 01/12/2020
** Description:
** Input:
** Output:
******************************************************/
```

Part 2. Questions (write answers in your Word/text file, section labeled "Part 2")

(A) You may encounter a compiler error while writing your program. If so, save and study it. If not, modify your program to cause the compiler to report an error.
(B) (**5 pts**) Copy and paste the compiler message in the "Part 2" section of your file.

(C) (**5 pts**) Use your detective skills (textbook, cplusplus.com, google, TAs, etc.) to learn what the error means. Write an explanation in your own words of what problem in the program caused the compiler to report this error.
(D) (**5 pts**) Write an explanation in your own words of how to fix the error.
(E) Fix the problem before you submit your code! Remember, submissions that do not compile receive a score of 0.

Part 3. Extra Credit (write answers in your Word/text file, section labeled "Part 3")

- (**up to 2 pts**) Ask another person (friend, family member, roommate, coffee shop stranger) to try out your fortune teller. In "Part 3" of your file, copy/paste the fortune they received. Did the program crash or behave in a way you did not expect? Describe anything you found surprising.
- (**up to 2 pts**) In your text file, describe one improvement you would recommend to make your program better (more robust, more entertaining, more attractive, anything).

Part 4. Submit Your Assignment Electronically

(A) Submit **your C++ program** (**.cpp** file, not your executable) and **your Word/text file with answers to Part 1, Part 2, and (optionally) Part 3** before the assignment due date, using TEACH.
(B) Remember to sign up with a TA on Canvas to demo your assignment.
The deadline to demo this assignment is 01/26/2020.