CSci 1113, Spring 2019
Lab Exercise 2 (Week 3): Sequence and Selection

Sequence refers to the idea that any computational process follows a specific order of execution. For example, in C++ each statement is completed prior to beginning any other statement, and operators within statements have an execution order based on operator precedence. *Selection* describes the alteration of the sequence based on dynamic conditions.

In this exercise you will continue to build your understanding of simple C++ programs, including those that modify processing based on simple dynamic selection. As usual, work in pairs, take advantage of all the help around you, and be sure to explore on your own.

Note: save the programs your and your partner write for the lab. They might be useful when you do future work in this class. Each problem solution should be maintained in its own (separate) source file. Moreover, both you and your partner should each have a copy of the programs in your individual accounts. Remember that, unlike the labs, the homework problems are to be done individually. You may reuse any code you and your partner wrote in lab, but if you do so the homework modifications you make should be done on your own, rather than collaboratively.

Lab 1 Cancellation/Make-up

If your lab got cancelled last week and you did not do all the lab problems on your own, then follow the instructions below; otherwise skip to the next section and start this week's lab.

1. If possible, find a partner who also did not finish last week's lab. But if this is not possible, then you can work with someone who has finished that lab.

2. You should have finished the Warm-up and Stretch problems for last week's lab. But if you have any questions about those, you may ask any of the TAs about those problems briefly at the beginning of today's lab.

3. If neither you nor your partner finished the workout problem on last week's lab, then do that problem first, before going to this week's problem. When you are done with it, have a TA check it, and then do as much of this week's lab as you can in the allotted time. You should still be able to get through most if not all of today's lab.

4. If one of you and your partner finished the workout problem on last week's lab, but the other did not, then (a) start by working together to do Warm-up Problem 1 and Stretch Problem 1 from this week's lab. (b) The person who previously finished last week's Workout problem should then do Stretch Problem 2 from this week's lab,

while the person who didn't finish last week's problem should work on it; so you'll be working separately at this time. If either of you finishes significantly before the other, then help the other person if they are stuck on any part of their problem. You can also ask the TAs for help if you get stuck. (c) Then, do Workout Problem 1 below together. (d) If you have time, the person who did not do Stretch Problem 2 can then go back and do that problem.

Warm-Up

(Warmup 1) Special Relativity

In special relativity, an object that has length L centimeters when at rest with respect to the observer has a relativistic length, L_R centimeters, given by

$$L_R = L\sqrt{1 - \frac{v^2}{c^2}}$$

when traveling at velocity v away from the observer. Here c is the speed of light, approximately 3.0×10^{10} cm/sec.

Write a program that will calculate and display the relativistic length L_R of an object of rest length L traveling at velocity v. In your program:

- Use object type `double` for L, v, c, and L_R.
- Initialize c to 3.0×10^{10} cm/sec.
- Have the user input values for L and v.
- Include the file `cmath` so you can use the `sqrt()` function.

Test your program using $L = 32.5$ cm and $v = 2.2 \times 10^{10}$ cm/sec as one of your test cases. When you enter the value of v use `e` (scientific) notation (see page 61, Section 2.3 of the textbook). Also test your program with $L = 42.0$ cm and $v = 3.2 \times 10^{10}$ cm/sec (again, enter v using scientific notation). What happens?

When you are done with this part, and understand the program behavior, then ask a TA to check your work.

Stretch

(Stretch 1) Basal Metabolic Rate (from the Savitch Ch 2 Programming Projects)

The Harris-Benedict equation estimates the number of calories your body needs to maintain your weight if you do no exercise whatsoever. This is called your basal metabolic rate, or BMR (note this is a single variable name, not three variables multiplied together). The formula for the calories needed for a woman to maintain her weight is

$$BMR = 655 + (4.3 \times \textit{weight in pounds}) + (4.7 \times \textit{height in inches}) - (4.7 \times \textit{age in years}).$$

The formula for the calories needed for a man to maintain his weight is

$$BMR = 66 + (6.3 \times \textit{weight in pounds}) + (12.9 \times \textit{height in inches}) - (6.8 \times \textit{age in years}).$$

A typical chocolate bar contains approximately 230 calories. Write a program that allows the user to input his or her weight in pounds, height in inches, age in years, and the character 'M' for male or 'F' for female. The program should then output the number of 230 calorie chocolate bars that need to be consumed to maintain one's weight for a person of the input sex, height, weight, and age.

Testing Discussion: You are your partner should separately come up with 3 sets of test data for this program. When both of you have made up your test cases, discuss with your partner why you chose the test cases you did, and, more generally, what makes good test cases for a program like this.

When you are done with this part, go on to the next part.

(Stretch 2) Temperature Conversion

For weather reporting, the daily temperature will generally be given in degrees Celsius or Fahrenheit. Write a C++ program that will convert a temperature given in one system of units to the other. For example, if the temperature is given in degrees Fahrenheit convert it to Celsius, and vice versa.

Your program should prompt the user to provide a temperature value (object type `double`) and a single *character* ('f' or 'c') to indicate if the value is in degrees Fahrenheit or Celsius. Read in the temperature and scale values, then compute and display the corresponding equivalent temperature in the other scale system. Your output message should indicate whether the result is Celsius or Fahrenheit. For example:

```
 Example 1:
              Enter the temperature:  26.6
              Enter Celsius (c) or Fahrenheit (f):  c
              The temperature in Fahrenheit is 79.88

Example 2:
              Enter the temperature: 93.2
              Enter Celsius (c) or Fahrenheit (f): f
              The temperature in Celsius is 34
```

Use the following conversion formulas:

$$\begin{aligned} F &= C \times (9/5) + 32 \\ C &= (F - 32) \times (5/9) \end{aligned}$$

[*Hint*: remember there is a difference between division of integers and division of doubles. Be careful when translating the equations above into C++.]

Test your program using the following values, as well as some other test cases you and your partner think up: (i) convert 37 degrees Celsius to Fahrenheit; (ii) convert 98.6 Fahrenheit to Celsius; (iii) convert 32 Fahrenheit to Celsius; (iv) convert -40 Fahrenheit to Celsius.

When you are done with this part, then ask a TA to check your work.

Workout

(Workout 1) Payroll (from a Savitch Chapter 2 programming project)

An employee is paid at the rate of $16.78 per hour for the first 40 hours worked in a week. Any hours over that are paid at the overtime rate of one-and-one-half times that. From the worker's total pay, 6% is withheld for Social Security tax, 14% is withheld for federal income tax, 5% is withheld for state income tax, and $10 per week is withheld for union dues. If the worker has three or more dependents, then an additional $35 is withheld to cover the extra cost of health insurance beyond what the employer pays. Your task is to write a program that will read in the number of hours worked in a week and the number of dependents as input, and will then output the worker's total pay, each withholding amount, and the net take-home pay for the week. Solve this problem in four steps:

(i) *Problem solving discussion*: Both you and your partner should work on it first individually (using a pencil and paper — don't start typing yet) for three minutes. This will not be enough time to solve the problem, but will be enough for you to check if you understand the problem, and to get started on solving it. After three minutes, compare the problem solving approach you are using with your partner's approach. Are they similar or different?

(ii) *Outline*: You and your partner should collaboratively come up with a detailed outline of the problem. Don't use C++ yet. Instead use an English-like description (pseudocode) to outline all the steps the program will need to do, in the order it will need to do them.

(iii) *Code*: Once you have the outline written, (collaboratively) turn it into a C++ program.

(iv) *Test and revise*: Make up test cases, test your program, correct it as needed, and continue testing and revising until your program is correct.

Check and Discussion

Before going on to the challenge question you and your partner should individually write down (i) one important thing you learned in the lab so far, and (ii) one question you still have about the C++ you used in the lab today. When you have both written down answers, share them with each other.

Before going on to the challenge questions, have a TA check your work.

Challenge

These problems are for those of you who just can't get enough! You should be able to complete the warm-up, stretch, and workout problems in the lab. Try this problem if you have extra time or would like additional practice outside of lab. If you complete either of both of these problems, you do not need to have a TA check your work (although you are welcome to do so if you wish and if a TA is free).

(Challenge 1) Ancient Greek Taxation

Suppose that in ancient Greece, where the unit of currency was the drachma, the following income tax code was used:

```
first 10,000 drachmas:  0% tax
next 20,000 drachmas:  10% tax
next 40,000 drachmas:  20% tax
drachmas after 70,000: 30% tax
```

For example, someone earning 100,000 drachmas would owe

$$10,000 \times 0.0 + 20,000 \times 0.1 + 40,000 \times 0.2 + 30,000 \times 0.3 = 19,000 \text{ drachmas.}$$

Write a program reads in an income value in drachmas, and outputs the amount of tax.

(Challenge 2) Selection Problem from Your Major

Think up a simple problem, from your major field, whose computer solution involves selection. Both you and your partner should think up separate problems. When you have each thought up a problem, share them and, if time permits, chose one and write a program that implements its solution.

Logout

Remember to logout of the computer before you leave.