COMP 1005/1405
Summer 2017 - Tutorial #3

Objectives

- Practice writing code using conditional/branching statements
- Practice writing code using loops
- Designing algorithms and solving more complex problems

Some hints for questions 2-5 are included at the end of the document in case you are struggling. Be sure to try each question first before reading them.

Problem 1 (Happiness Check)

A person who is wet and cold is miserable. A person who is poor and hungry is miserable. Everyone else is happy. Write a program that asks the user for the required information and determines if they are miserable or happy. Consider using nested if statements without logical operators, as well as a single if/else statement using logical operators.

Problem 2 (Divisors and Prime Numbers)

a) Write a program that uses a while loop to print all divisors of a number supplied by the user. The program should also print the sum of all the divisors and whether the number the user entered is a prime number or not. Note: The definition of a divisor is a number that divides another evenly (i.e., without a remainder) and the definition of a prime number is a number whose only divisors are 1 and itself.

b) Implement the same program as above using a for loop instead of a while loop. Which implementation do you think is the better choice for this problem?

Sample Outputs (user input highlighted)

Enter an integer: 20
The divisors are:

1
2
4
5
10
20
The sum of the divisors is 42
The number is not prime.

Enter an integer: 19
The divisors are:
1
19
The sum of the divisors is 20
The number is prime!

Problem 3 (Garden Sensor)

You have a prized vegetable garden in your backyard, but are worried about the water levels. You work a significant distance from your house and are unsure if it rains on your garden during the day, so you do not know if your garden needs watered or not. You install a rain sensor in your garden, which can answer the True/False question of whether it rained during the day. As you are a busy person, you cannot check the sensor each day, so you must write a program that will print out a warning message and stop sensing if it does not rain for 3 days in a row. Note: to keep tuition below $100,000/year, we do not have a real garden and sensor, so you will be responsible for typing in the sensor measurements (True or False) for the program.

Sample Output

It rained today? True
It rained today? True
It rained today? False
It rained today? True
It rained today? False
It rained today? False
It rained today? False
Quick! Water your garden before all the vegetable plants die and you starve to death!

Problem 4 (Increasing Sequences)

This problem and problem 5 are more difficult than the others, but are still worth giving a shot. Just remember to break the problem down, identify what values you need to keep track of to meet the program requirements and determine how these values need to be modified based on the inputs of the user. It is suggested to produce and verify an algorithm on paper before starting to code.

An increasing sequence of numbers is one in which each number is larger than the one before it. For example, [4, 8, 13, 14, 21] is an increasing sequence, but [4, 8, 13, **12**, 21] is not. Write a program that repeatedly reads positive integers from the user. Your program should track the length of the longest increasing sequence of numbers that the user has entered and display it when the user has finished entering numbers. The user is finished entering numbers when they input “q” or when they enter a decreasing sequence of numbers (i.e., each number is lower than the previous) of length 3.

Sample Outputs (user input highlighted and longest sequence bolded)

Enter a positive integer or 'q' to quit: **1**
Enter a positive integer or 'q' to quit: **3**
Enter a positive integer or 'q' to quit: **4**
Enter a positive integer or 'q' to quit: **5**
Enter a positive integer or 'q' to quit: 4
Enter a positive integer or 'q' to quit: 6
Enter a positive integer or 'q' to quit: 5
Enter a positive integer or 'q' to quit: 4
Length of longest increasing sequence is 4

Enter a positive integer or 'q' to quit: 1
Enter a positive integer or 'q' to quit: 5
Enter a positive integer or 'q' to quit: **3**
Enter a positive integer or 'q' to quit: **6**
Enter a positive integer or 'q' to quit: **7**
Enter a positive integer or 'q' to quit: 5
Enter a positive integer or 'q' to quit: 8
Enter a positive integer or 'q' to quit: 6
Enter a positive integer or 'q' to quit: q
Length of longest increasing sequence is 3

Problem 5 (Distance Calculator)

Write a program that repeatedly reads x and y coordinates representing the location of cities. Each time a new city is entered, the program should calculate the Euclidean distance between

the last city entered and the new city (i.e., how far it would be to travel between the two cities). The program should track the total distance and display it once the user enters "q" for either the x or y coordinate. Your program can assume that the user will always enter integer values. To calculate the square root of a value, you can import the math library (`import math`) and use the square root function (`math.sqrt(`*`some_number`*`)`).

Note, if city #1 is at location (x1,y1) and city #2 is at location (x2,y2), the Euclidean distance between them can be calculated as:

$$dist = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

Hints and Extra Information

Problem #2 Hint #1: A prime number is divisible by itself and 1, which means it has 2 divisors in total.

Problem #2 Hint #2: You have determined all of the divisors of the number, all you have to do is see if the number of divisors you found is equal to 2. This will allow you to say whether the number is prime or not.

Problem #3 Hint #1: We discussed this problem in class previously. To determine if the alarm should go off, all you need to determine is if it has been 3 days without rain or not. This implies that you need to count the number of days since it rained.

Problem #4 Hint #1: Any time the user enters a number, there are two possible cases: The number extends an increasing sequence OR the number is the start of a new increasing sequence

Problem #4 Hint #2: You need to count the length of the current increasing sequence and, when it ends, determine if it was longer than the longest increasing sequence seen so far. The two cases above will allow you to decide which action to take.

Problem #5 Hint #1: To calculate the increase in distance, you must have access to the x/y coordinates of both the city that was just entered by the user (call it new_city) and the previously entered city (call it old_city).

Problem #5 Hint #2: When the user enters the x/y coordinates of the first city, the distance is just 0, so you do not need to calculate anything. You can, however, immediately initially ‘old city’ variables to store the x/y coordinate of the first city. When another city is entered and stored in the ‘new city’ variables, you can calculate the distance and then update the values of the ‘old city’ variables.