02.03 Input & Selection Structures

Name:

Directions

You will follow the four steps in computational thinking to develop and design this program.

Step One: Decomposition

Decomposition involves identifying a problem and breaking it down into smaller, more manageable steps.

Example: A retailer thinks its online sales should be higher. They identify that their customers are not making as many large purchases online compared to in the physical store. Looking even closer, they identify a lack of $200 and higher purchases.

Step Two: Pattern Recognition

After the problem has been broken down in the decomposition step, additional data is needed.

Example: The retailer recognizes a pattern in online sales. When the purchase price is $200 or more, the shipping costs are also quite high, making customers less likely to make these purchases online. Perhaps an incentive is needed?

Step Three: Algorithmic Design

Here's where the fun (and the assignment) really begins. Choose one of the options provided below:

> **Option 1: Spend More, Get More**
> You've been hired by a store to write a program that helps improve online sales. The retailer wants to find out how much money customers want to spend, offer a recommendation, and encourage customers to spend at least $200 by offering a free gift at this price point.
>
> First, you should ask the customers how much money they would like to spend. Then recommend a product that can be purchased within the amount entered. Finally, let the customers know how much more money they need to spend in order to receive a free gift (or let them know they qualify if their initial purchase is $200 or above).

Option 2: Choose Your Own Adventure
Do you enjoy being creative and making up your own scenarios? If you do, then this option is for you! Come up with any scenario for this assignment as long as you do the following:

- Make it shopping-related (this includes delivery, sales, accounting, or anything else that works directly with retailers).
- Ensure people reading your code understand your scenario (Hint: Use lots of comments in your code).
- Follow the requirements described below.

Algorithmic Design and Pseudocode

Use the algorithmic design stage of computational thinking to design a program. Remember that algorithmic design is where you create step-by-step instructions to solve a problem. This must begin with pseudocode, then translate to Python using your pseudocode as a guide.

Write your pseudocode where indicated below. Your pseudocode should be written in English (not Python), should indicate a program that is improving the retail industry, and must:

be written in a series of steps that reflect algorithmic design

print a description of your online store for the user to read

obtain user input in order to make a recommendation

demonstrate proper use of the `int()`, `float()`, and `str()` functions

use proper order of operations and appropriate math functions for calculations

include at least one nested if-else/elif statement

provide output based on user input and results of calculations

Example of expected output: The output for your program should resemble the following screenshot. Your specific results will vary depending on the choices you make and the input provided.

Pseudocode Hint: Pseudocode for the example output above began with "Start" (which represented the main function). The next line of pseudocode read "Store Description" (which later turned into a print function to inform the user about the store). Then the pseudocode read "Ask how much the customer wants to spend" (this was the eventual input), etc. When you're finished writing your pseudocode, ask yourself if a non-programmer would be able to read it and understand what your program is going to do. If the answer is yes – you've probably done a

great job!

Write your pseudocode here:

Pseudocode to Program Code

Use the Python IDLE built into this course to code and run your program. Your code must:

> Use comments for internal documentation (including a heading with your name, today's date, and a short description of the program). Remember that you can copy/paste much of your pseudocode for this internal documentation.
>
> Follow the Python style conventions regarding indentation and the use of white space in your program.
>
> Run successfully and produce output similar to the example above.

When you've completed writing your program code, save your work by selecting 'Save' in the Python IDLE. When you submit your assignment, it will be in two files:

1. Your Python code (as a .py file)
2. This completed assignment template

Step Four: Generalize & Assess

Complete the Post Mortem Review (PMR). Write thoughtful two- to three-sentence responses to all the questions in the PMR chart.

Post Mortem Review Question	Response
What was the purpose of your program?	
How could your program be useful in the real world?	
What is a problem you ran into, and how did you fix it?	
Describe one thing you would do differently the next time you write a program.	

How could your program be generalized and useful in other areas?	