SCALABLE DATA INFRASTRUCTURES
full sail university

SDI

Activity: Logic & Loops

Before You Begin:

Go thoroughly read the SDI_LogicLoops.pdf! This contains the full instructions for this project and the rubric, which explains how you will be graded.

Criteria:

For each problem below you will need the following:

- Label the section of code appropriately
- Prompt the user for each variable that is in the “User Input” Section of that problem.
- Validate each user prompt with a **while loop** to insure that the user is typing in a valid response. Remember we check text string different than numbers!
- Convert each user response to the **correct** data type, if needed.
- The result should be calculated using variables, not literal values when possible.
- Create code that will make decisions based on the value of the user’s responses.
- Print the result of the decision-making to the console using the format given in the “Results” section of that problem.
- After each section put in a multi-lined code for the Test Values.
- Use only code and techniques learned in this class.

Graded Problems:

Problem #1 – Logical Operators: Tire Pressure I

To meet to maintenance standards a car's front two tires should have the same pressure and the back two tires should have the same pressure. However the front tires and the rear tires can have different pressure than each other, so it is not necessary for all four tires' pressure to be the same!

Create a **single** conditional that would determine if the tires of a given car are up to spec. Make sure to use the array value inside of the conditional.

- **User Inputs:**
    - Prompt for the pressure in each of the (4) tires one at a time.
        - Store the converted number values inside of an array that you create.
- **Result To Print Out:**
    - "The tires pass spec!" Or " Get your tires checked out!"
- **Data Sets To Test:**
    - Front Left – 32, Front Right – 32, Back Left -30 Back Right- 30 - Tires OK
    - Front Left – 36, Front Right – 32, Back Left -25 Back Right- 25 - Check Tires
    - Test One Of Your Own & Write the results in your comment section.

Problem #2 – Logical Operators: Movie Ticket Price

The local movie theater in town has a normal ticket price of **$12.00**.

If you are a senior (55 and older) or you are under 10, you get the discounted price of **$7.00**.

In addition, if a customer is seeing a movie between 14 and 17 aka (2pm- 5pm), you get the discounted price of **$7.00**.

Create a **single** conditional block that would determine which of the two prices the customer is eligible for. Make sure you are using logical operators!

- **User Inputs:**
    - Age of the customer
    - Time of Movie (Assume whole numbers here and use military time.)
- **Result To Print Out:**
    - "The ticket price is $**X.**"
- **Data Sets To Test:**
    - Age – 57, Time – 20, Ticket Price - $7.00
    - Age – 9, Time – 20, Ticket Price - $7.00
    - Age – 38, Time – 20, Ticket Price - $12.00
    - Age – 25, Time – 16, Ticket Price - $7.00
    - Test One Of Your Own & Write the results in your comment section.

Problem #3 – For Loop: Add Up The Odds or Evens

Create an array of 6 or more integers. Then, you will prompt the user if they would like to see the sum of the even numbers or odd numbers from the array.

For this question you do NOT have to ask the user for the values inside of the array, these can be hardcoded.

Use a FOR loop to cycle through the array and find the correct sum that the user wants.

*Hint- If you need help finding if it is odd or even, review the Modulus operator.

- **User Inputs:**
    - Prompt the user to either see the sum of the odd or even numbers.
- **Result To Print Out:**
    - "The odd numbers add up to **X**" or "The even numbers add up to **X**"
- **Data Sets To Test:**
    - Array: {1,2,3,4,5,6,7}, Sum of Evens – 12 Sum of Odds – 16
    - Array: {12,13,14,15,16,17}, Sum of Evens – 42 Sum of Odds – 45
    - Your array should be different than mine, make sure to test both even and odd and put your results in this section.

Problem #4 – While Loop: Charge It!

While loops are an excellent choice when you need code to repeat an unknown number of times.

In this problem you are going to keep buying items until you reach your maximum credit limit. You will ask the user for their credit limit first.

Then create a while loop that will start deducting each purchase from the credit limit. As long as the credit limit is above zero, the while loop should continue looping. When the user goes over their credit limit, stop the loop and tell them how much they went over by.

- **User Inputs:**
    - Max Credit Limit
    - Inside of the loop, keep asking the user how much their purchase is.
- **Result To Print Out:**
    - For each "purchase" (hint: *every time it loops*) the loop should tell the user how much money they have left that they could spend. "With your current purchase of $**X**, you can still spend $**X**."
    - After the user exceeds their limit, "With your last purchase you have reached your credit limit and exceeded it by $**X**."
- **Data Sets To Test:**
    - Credit Limit – 20.00
        - Purchase1- 5.00 - You can still spend $15.00
        - Purchase2- 12.00 - You can still spend $3.00
        - Purchase3- 7.00 - You have exceeded your limit by $4.00.
    - Test One Of Your Own & Write the results in your comment section.