AmazingDevOps

Work Due next week Thursday.

Project Title: Weekly Paycheck Calculator

Project Description: You are tasked with creating a program that calculates the weekly paycheck for an employee based on the number of hours they worked. The program should follow the specific payment rules of the company:

- The employee earns $20 per hour for the first 40 hours worked.
- For any hours worked above 40, the employee earns $30 per hour as overtime pay.

Example: If an employee works 60 hours in a week, they would earn $20 per hour for the first 40 hours and $30 per hour for the 20 hours of overtime. Therefore, their total earnings would be calculated as follows: ($20 per hour * 40 hours) + ($30 per hour * 20 hours) = $800 + $600 = $1400.

Instructions for the Project:

1. Set up the project:
    - Create a new Python project with an appropriate name.
    - Create a Python file to write your code.
2. Get input from the user:
    - Ask the user to enter the number of hours worked by the employee in a week.
    - Store the entered value in a variable.
3. Calculate the paycheck:
    - Define a constant variable **regular_rate** and assign it the value of $20 (the regular pay rate per hour).
    - Define a constant variable **overtime_rate** and assign it the value of $30 (the overtime pay rate per hour).
    - Define a constant variable **regular_hours** and assign it the value of 40 (the number of regular hours in a week).
4. Calculate earnings:
    - Use an **if** statement to check if the number of hours worked is greater than **regular_hours**.
    - If the condition is true, calculate the total earnings as follows:

        - Calculate the regular earnings: **regular_earnings = regular_rate * regular_hours**
        - Calculate the overtime earnings: **overtime_earnings = overtime_rate * (hours_worked - regular_hours)**
        - Calculate the total earnings: **total_earnings = regular_earnings + overtime_earnings**
    - If the condition is false, calculate the total earnings as follows:
        - Calculate the total earnings: **total_earnings = regular_rate * hours_worked**

5. Display the result:
    - Print the total earnings for the employee in a user-friendly format.
6. Test your program:
    - Run the program and test it with different input values.
    - Verify that the program calculates the paycheck correctly based on the number of hours worked.
7. Enhance the program (optional):
    - Add error handling to handle invalid input (e.g., negative hours or non-numeric input).
    - Allow the user to calculate paychecks for multiple employees or multiple weeks.
    - Implement a loop to continuously calculate paychecks until the user decides to exit the program.
8. Submit your project:
    - Submit your Python file or the entire project, depending on the submission requirements.

Note: Provide clear instructions and include comments in your code to explain the different steps and calculations.