Homework 01 – Introduction to Python

Overview

This homework will help you practice fundamental Python programming concepts introduced during Week 1, including data types, variables, basic arithmetic operations, and simple input/output. Completing these exercises will build your confidence writing Python code and prepare you for upcoming topics.

Instructions

- Write Python code to solve each problem below.
- Include comments in your code explaining your logic.
- Use meaningful variable names.
- Test your code before submission to ensure it runs without errors.
- Submit all your code in a single `.py` file or multiple files if you prefer (please clearly label each file).

Problems

Problem 1: Hello, Python!

Write a Python program that prints the message “Hello, Python World!” to the screen.

Hint: Use the `print()` function.

Problem 2: Variables and Data Types

Create variables to store the following information about yourself:
- Your full name (string)
- Your age (integer)
- Your height in meters (float)
- Whether you are a student (boolean, True or False)

Print each variable with a descriptive message. For example:
`print("My age is:", age)`

Problem 3: Simple Arithmetic

Write a program that:
- Asks the user to input two numbers (use `input()`)

- Converts the inputs to floats
- Calculates and prints their sum, difference, product, and quotient

Hint: Use `float()` to convert strings to numbers.

Problem 4: String Manipulation

Write a Python program that:
- Asks the user for their first name and last name separately
- Combines them into a full name
- Prints a greeting message such as “Hello, John Smith!”

Hint: Use string concatenation with the `+` operator or f-strings.

Problem 5: Data Type Identification

Given the following variables:
```python
a = 10
b = 3.14
c = "Python"
d = True
```
Write code that prints the value and data type of each variable.

Hint: Use the `type()` function.

Problem 6: Bonus: Basic Calculator Function

Define a function called `calculator` that:
- Takes two numbers and a mathematical operator (string: '+', '-', '*', '/') as parameters
- Returns the result of the operation on the two numbers
- Prints an error message if the operator is not recognized

Test your function with example inputs.

Problem 7: Temperature Converter

Write a Python program that:
- Asks the user to input a temperature in Celsius.
- Converts it to Fahrenheit using the formula: `F = (C * 9/5) + 32`
- Prints the result with a clear message.

Problem 8: Age Calculator

Write a program that:
- Asks the user to input their birth year.
- Calculates their age assuming the current year is 2026.
- Prints out their age with a descriptive message.

Problem 9: Even or Odd

Write a Python program that:
- Asks the user for an integer.
- Determines if the number is even or odd.
- Prints an appropriate message.

Problem 10: Simple Interest Calculator

Write a program that:
- Takes principal amount, rate of interest (annual), and time (in years) as input from the user.
- Calculates simple interest using the formula: `SI = (P * R * T)/100`
- Prints the calculated interest.

Problem 11: Circle Properties

Write a program that:
- Asks the user for the radius of a circle (float).
- Calculates and prints the circumference and area of the circle.
- Use `3.1416` for π.

Problem 12: Swap Two Variables

Write Python code to:
- Take two numbers as input from the user.
- Swap their values without using a temporary variable.
- Print the values before and after swapping.

Problem 13: List of Favorite Colors (Intro to Lists)

Write a program that:
- Creates a list of your top 5 favorite colors.
- Prints the list.
- Prints the first and last color from the list.

Problem 14: User Info Summary

Write a program that:
- Takes user inputs for name, age, and city.
- Prints a formatted summary: “Name is Age years old and lives in City.”

Helpful Tips and Reminders

- Use comments (`#`) to explain your code.
- Use consistent indentation (4 spaces is standard).
- Test each part of your code separately to debug easily.
- Refer to Week 1 lecture slides and the official Python documentation if needed.
- You can use online Python interpreters like Repl.it if you don’t have an IDE set up yet.

Submission Checklist

- [ ] Python script file(s) submitted to ELMS Canvas.
- [ ] Code includes comments explaining your logic.
- [ ] All problems completed and tested.