Tutorial I - Introduction

Programming with Python

Introduction

Solutions

You will likely find solutions to most exercises online. However, I strongly encourage you to work on these exercises independently without searching for answers. Understanding someone else's solution is very different from developing your own. Use the lecture notes and try to solve the exercises on your own. This approach will significantly enhance your learning and problem-solving skills.

Remember, the goal is not just to complete the exercises, but to understand the concepts and improve your programming abilities. If you encounter difficulties, review the lecture materials, experiment with different approaches, and don't hesitate to ask for clarification during class discussions.

How to tackle the exercises

I would recommend the following approach: Create a separate `.py` file for each exercise. First, try to understand the problem and what it is asking you to do. Then, write the code to solve the problem. Run the code and check if the output is correct. If it is not, try to find a solution together with the instructor or classmates.

Tutorials vs Assignments

In the tutorials, we will go through different exercises together and you can ask questions. The assignments are similar to the tutorials, but you will work on them individually or in small groups. You can still ask questions during the tutorial sessions, but you should try to solve the exercises on your own first.

Basic String Operations

In this exercise, we'll explore basic string operations in Python. Strings are one of the most common data types in programming, and Python provides a variety of ways to manipulate them. We'll practice concatenation (joining strings together), indexing (accessing individual characters), slicing (extracting portions of a string), finding the length of a string, and string repetition.

```
# a) Concatenation
# TODO: Concatenate "Coffee" and " is fun!" in the 'text' variable
# Print the result
# Your code here

# b) Length
```

```
# TODO: Print the length of 'text'
# Your code here

# c) Indexing
# TODO: Print the first and last character of 'text'
# Your code here

# d) Slicing
# TODO: Print the first three characters of 'text'
# Your code here

# e) Repetition
# TODO: Print 'Coffee' repeated 3 times
# Your code here
```

Arithmetic Operations

In this exercise, we'll practice using arithmetic operations in Python to solve several real-world problems. These calculations will help you understand how to apply basic mathematical operations in programming and how to use variables to store and manipulate numerical data. We'll cover temperature conversion, geometric calculations, financial computations, and time calculations.

```
# a) Temperature Converter
# TODO: Convert 98.6°Fahrenheit to Celsius
# Formula: (°F - 32) * 5/9
# Your code here

# b) Circle Area Calculator
# TODO: Calculate the area of a circle with radius 5
# Use 3.14159 for π
# Your code here

# c) Simple Interest Calculator
# TODO: Calculate the average interest
# Principal of 1946, rate of 5 % per year, 3 years
# Your code here

# d) Time Duration Calculator
# TODO: How many minutes and seconds are in 298471 seconds
# Your code here (use floor division and modulo)

# e) Compound Growth
# TODO: Calculate the final amount after 3 years
# You invest 23842 with 8% annual growth
# Formula: initial_amount * (1 + growth_rate)^years
# Your code here
```

BMI Calculator

In this exercise, we'll create a simple Body Mass Index (BMI) calculator. BMI is a measure that uses your height and weight to work out if your weight is healthy (although it has

its limitations). The BMI is calculated by dividing a person's weight (in kilograms) by the square of their height (in meters). This exercise will reinforce your understanding of arithmetic operations, user input, and string formatting while introducing you to a real-world health metric calculation.

```
# TODO: Implement a BMI calculator
# - Ask for weight in kg
# - Ask for height in meters
# - Calculate BMI
# - Print the result rounded to two decimal places
# - Make use of f-strings for the output
# Hint: You can use input() to get the weight and height and remember the
type of the variables!
# Your code here
```

Formatted Output

In this exercise, we'll practice using formatted output in Python. Formatted output allows us to present information in a structured and visually appealing way. We'll use f-strings (formatted string literals) to create a simple receipt for a coffee shop purchase. This exercise will help you understand how to align text, format numbers, and create a clean, readable output.

> Tip
>
> Use the `f-string` method to format the output. New is the `f"{your_variable:.2f}"` syntax, which you can use to format the variable to two decimal places and the `f"{your_variable:<20}"` syntax, which you can use to format the variable to left-aligned within 20 spaces.

```
# Create a receipt for a coffee shop purchase
# Use the following information:
item1 = "Cappuccino"
price1 = 3.50
item2 = "Blueberry Muffin"
price2 = 2.25
item3 = "Bottled Water"
price3 = 1.50

# a) TODO: Calculate the subtotal
# Your code here

# b) TODO: Calculate the tax (assume 8% tax rate)
tax_rate = 0.08
# Your code here

# c) TODO: Calculate the total
# Your code here
```

```
# d) TODO: Print the receipt
# It should look similar to this:
#
# ===== Coffee Shop Receipt =====
# Cappuccino        $  3.50
# Blueberry Muffin  $  2.25
# Bottled Water     $  1.50
# --------------------------
# Subtotal          $  7.25
# Tax (8%)          $  0.58
# --------------------------
# Total             $  7.83
# ========= Thank You! =========
#
# Make sure to align the prices to the right
# Format the currency with 2 decimal places.
# Your code here
```

That's it!

After a week, you can find the solutions to these exercises online in the associated GitHub repository, but we will also quickly go over them in next week's tutorial. To access the solutions, click on the Github button on the lower right and search for the folder with today's lecture and tutorial. Alternatively, you can ask ChatGPT or Claude to explain them to you. Remember, the goal is not just to complete the exercises, but to understand the concepts and improve your programming abilities.