Unleashing the Superpowers of Functions: An Experiential and Active Learning Journey – TA Guide

ENDG 233 – Programming with Data

University of Calgary

The Schulich School of Engineering (SSE)

Shadi Aljendi

June 27, 2023

Dear TA,

Thank you for assisting in the upcoming session of the Flipped Learning Module on Functions in Programming for the ENDG 233 course. This guide aims to provide you with an overview of the challenges, potential questions, and corresponding answers that students may encounter during the activities. Please familiarize yourself with the following information to effectively support the students during the session.

Challenges

Approach the challenges in an experiential manner by encouraging exploration, critical thinking, problem-solving discussions, and providing guidance without immediate answers. Here are some general recommendations.

1. Facilitate a discussion: Encourage students to analyze the problem requirements, review the available functions, and discuss their understanding of the task. Prompt them with questions that guide them toward the correct functions and parameters.
2. Provide hints and examples: Offer hints that nudge students in the right direction without giving away the complete solution. Provide examples of correct syntax or demonstrate how to use the functions correctly. Demonstrate step-by-step execution: Show students how to execute the code step by step, highlighting the flow of control and how the functions interact with the main program. Use visuals or diagrams if necessary.
3. Encourage testing and debugging: Emphasize the importance of testing the code with different inputs to verify its correctness. Guide them on how to debug their code by using print statements or a debugger to trace the execution and identify any issues.
4. Offer individual assistance: If students are still struggling, provide one-on-one assistance by reviewing their code, identifying errors, and providing targeted guidance. Help them troubleshoot and iterate on their solutions.
5. Reinforce good programming practices: Remind students to follow good programming practices, such as using meaningful variable names, proper indentation, and clear documentation. Discuss the importance of code readability and maintainability.

6. Encourage peer collaboration: Facilitate peer-to-peer collaboration by suggesting students work together, share ideas, and provide feedback to each other. Encourage them to discuss their approaches and learn from one another.

Filling in the Blanks (Activity 1)

If students face challenges in identifying the appropriate functions and ensuring correct syntax and input/output behaviour, you can support them by:

1. Reinforcing function definition basics: Remind students about the fundamental concepts of function definitions, including the use of "def" and "return" statements.
2. Exploring function signatures: Assist students in understanding the purpose and parameters of the functions by reviewing their signatures. Discuss the expected input and output of each function and how they contribute to solving the overall problem.

Creating a Function (Activity 2)

Students may need guidance on defining function parameters, accessing and manipulating input values, and returning the converted temperature correctly.

1. Walk through step-by-step: Break down the process of defining function parameters, accessing input values, and performing necessary calculations. Guide students through each step to ensure they understand how to correctly implement these actions.
2. Highlight common mistakes: Point out common errors or misconceptions that students may encounter when working with function parameters, input values, or return statements. Explain why these mistakes are problematic and offer strategies to avoid them.

Modular Programming (Activity 3)

Students may face challenges in identifying code segments to encapsulate into functions, defining function signatures, and integrating the functions into the main program.

1. Analyze the code together: Review the provided code and identify potential segments that can be encapsulated into functions. Discuss the purpose and functionality of each segment and how it contributes to the overall program.
2. Guide function definition: Assist students in defining function signatures by discussing the necessary parameters and return values. Help them understand the inputs required for each function and how they relate to the specific problem.

Extra Challenge Activity: Composite Fibonacci Numbers

Students may require assistance in generating the Fibonacci sequence, checking for compositeness, and efficiently filtering and printing the desired numbers.

1. Review the Fibonacci sequence: Discuss the concept of the Fibonacci sequence and how it is generated. Help students understand the pattern and how each number is the sum of the two preceding numbers.
2. Guide in generating the sequence: Explain the logic behind generating the Fibonacci sequence and provide examples or pseudocode to illustrate the process. Assist students in implementing the code to generate the sequence within the desired range.

3. Provide examples and test cases: Share examples of the expected output for different input values. Provide test cases that cover different scenarios and edge cases to help students verify the correctness of their implementation. Provide examples and test cases: Share examples of the expected output for different input values. Provide test cases that cover different scenarios and edge cases to help students verify the correctness of their implementation.

Testing and Debugging

Students may seek guidance on effective testing and debugging strategies within their chosen Integrated Development Environment (IDE).

Time Management

Students may require advice on managing their time effectively to complete the activities within the given time frame.

Common Questions and Answers:

How do I fill in the blanks in Activity 1?

Answer: Analyze the provided code and determine the missing function calls and parameters. Replace the blanks with the correct function names and variables.

Can I use the same approach for Activity 2 as in Activity 1?

Answer: No, Activity 2 requires creating a single function that handles both temperature conversions. Think about how you can combine the logic from both functions in Activity 1 into a single function.

How can I modularize the code in Activity 3?

Answer: Identify code segments that can be encapsulated into functions. Create a function that tests if a number is even or odd and use it within the main program to count the even and odd numbers.

What is the best approach to solve the Extra Challenge Activity?

Answer: There are multiple approaches to solving the Extra Challenge Activity. Discuss different strategies with your group or individually, such as generating the Fibonacci sequence and checking for compositeness.

Are there alternative approaches or optimizations for these activities?

Answer: Yes, there are often multiple ways to solve a programming problem. Discuss different approaches and optimizations with your group or individually. Consider trade-offs between efficiency, code complexity, and maintainability

Please remember that as a TA, your role is to guide and support the students in their learning journey. Encourage active participation, critical thinking, and problem-solving skills. Provide hints, examples, and guidance when necessary, but allow students to explore and discover solutions on their own.

If you have any further questions or require additional guidance, please don't hesitate to reach out to me. Thank you for your valuable contribution to the success of the session.

Best regards.
Shadi