Python Programming

Assignment 1- Coding with and without functions

Python Program- Quarter of the Year code:

With Functions	Without Functions

def determine_quarter(month):
  if 1 <= month <= 3:
  return "First quarter"
  elif 4 <= month <= 6:
  return "Second quarter"
  elif 7 <= month <= 9:
  return "Third quarter"
  elif 10 <= month <= 12:
  return "Fourth quarter"
  else:
  return "Error: Invalid month"
 	
user_input = int(input("Enter a month (1-12): "))
print(determine_quarter(user_input))
	user_input = int(input("Enter a month (1-12): "))
	if 1 <= user_input <= 3:
	  result = "First quarter"
	elif 4 <= user_input <= 6:
	  result = "Second quarter"
	elif 7 <= user_input <= 9:
	  result = "Third quarter"
	elif 10 <= user_input <= 12:
	  result = "Fourth quarter"
	else:
	  result = "Error: Invalid month"
	
	print(result)

Comparison and observation:

The code with functions shows a more structured and modular design. The function ‘determine_quarter’ summarizes the logic for determining the quarter based on user input. This not only improves code readability but also promotes reusability. By separating the quarter-determining logic into a function, the code becomes more maintainable and scalable. This approach sticks to the principles of clean code, making it easier for developers to grasp and modify specific functionalities without affecting the entire program. The function acts as a separate module, providing a clear and concise way to handle the quarter-of-the-year problem.

On the other hand, the version without functions takes a more direct and linear approach. It simplifies the program by including the quarter-determining logic directly into the main code. While this might be suitable for smaller projects or quick implementations, it lacks the organizational benefits offered by functions. The absence of modularization may lead to code duplication in larger programs, making it harder to manage and maintain over time. However, for a concise and straightforward task like this, the non-function-based approach is practical and easily comprehensible.

Comparing both methods reveals a trade-off between simplicity and organization. The function-based approach excels in terms of code organization, readability, and reusability in critical aspects for larger projects. On the flip side, the non-function-based approach prioritizes simplicity, making it suitable for smaller tasks with less complexity. In the end, the choice between these approaches depends on the specific needs of the project, with functions offering a powerful tool for managing complexity and promoting code clarity, while the non-function approach is a quick and straightforward solution for simpler tasks.

References:

Python - Functions
W3Schools