Name: Cano, Marc Allen E. Code: 7599

Course: CPE 223/L

Questions:

1. What are the main differences between procedural and object-oriented programming?

The methodical process of procedural programming divides code into functions, putting a focus on following directions exactly. It handles information and processes as distinct beings. However, object-oriented programming (OOP) is focused on objects that, by encapsulating behavior and data, combine them. As opposed to OOP encourages modularity and code reuse in procedural programming by organizing applications that make use of objects and classes. Furthermore, OOP upholds important concepts like inheritance and polymorphism, which increases its adaptability and maintainability in contrast to, as a project expands, procedural programming can become burdensome. complexity.

2. How does OOP improve code reusability?

Through the use of key ideas like inheritance, polymorphism, abstraction, and encapsulation, OOP improves code reusability. By utilizing these ideas, developers can create modular, scalable, and reusable code that can be readily expanded without rewriting significant portions of a program. Inheritance allows new classes to inherit properties and behaviors from existing ones, reducing redundancy; polymorphism allows a single interface to handle different types, promoting flexibility in code design; and encapsulation makes sure that data is bundled with pertinent methods, preventing unintended modifications and enhancing maintainability.

Observation

In software development, the two paradigms have different functions. Simple, linear applications with a clear execution flow are a good fit for procedural programming. However, OOP is perfect for bigger, more intricate projects that need improved long-term planning, scalability, and ability to be maintained. The needs of the project and the team will determine which of these approaches is best. cooperation as well as the degree of adaptability needed for future growth.