Omówienie zadania 1 z Zasady SOLID

Ogólnie kod nie spełnia zasad SOLID, nie można go w łatwy sposób edytować poprzez dodawanie nowych funkcji programu.

4a. Dodano nową klasę Ryż, zmodyfikowano klasę Składnik, Zupa oraz klasę Test, tak, aby móc zamienić Makaron na Ryż. – Złamana zasada **Open-Close**.

4b. Dodano nową klasę ZupaNaWynos, która dziedziczy po klasie Zupa i zmienia działanie metody podaj.

4c. Dodano nową klasę Pomidorowa, która dziedziczy po klasie Zupa i zmienia metodę przygotuj, dodając nową metodę zainicjuj uruchamianą wewnątrz metody przygotuj.

4d/5. W tym momencie nie ma możliwości utworzenia w prosty sposób klasy PomidorowaNaWynos – Brak zasad **Single Responsibility** oraz **Dependency Inversion**. Należy utworzyć dwie klasy np. Kucharz i Kelner, jedna klasa będzie odpowiadać za gotowanie Zupy, druga będzie odpowiadać za jej podanie/dostarczenie.

Następnie, aby w prosty sposób wprowadzać kolejne implementacje zup, utworzono klasę Rosół, która dziedziczy po Zupie. Klasę Zupa tak zmodyfikowano, aby była klasą abstrakcyjną dla kolejnych klas. Na tej podstawie utworzono nową klasę Zur, która dziedziczy po klasie Zupa – konieczne do zachowania Zasady **Liskov Substitution Principle (LSP).**

Dodanie kolejnej funkcjonalności to utworzenie interfejsów, gdzie klasa Kelner implementuje interfejs Podawacz oraz nowa klasa Kurier implementuje interfejsy Podawacz oraz Adresowalny (konieczność podania posiłku pod wskazany adres). Wprowadzenie tych modyfikacji pozwala na zastosowanie zasady **Interface Segregation**.

W interfejsie Adresowalny dodatkowo ustawiona setera, tak aby zachować bezpieczeństwo pół w poszczególnych klasach i ich modyfikacja oraz pobieranie było możliwa tylko poprzez getery i setery.