SOLID

- **Single responsibility:** SPEŁNIONE, każda klasa jest przypisana do pełnienia jednej, konkretnej funkcji.

- **Open/closed:** SPEŁNIONE, istnieje możliwość rozbudowania programu bez modyfikacji jego sposobu działania w fundamentalny sposób. Użycie polimorfizmu - spełnione.

- **Liskov substitution:** SPEŁNIONE, można zawsze użyć dowolnej klasy pochodnej. Zachowana jest zgodność metod i interfejsu.

- **Interface segregation:** SPEŁNIONE, klasy są odpowiednio małe, a metody w środku klas dziedziczących są intuicyjne w użyciu.

- **Dependency inversion:** SPEŁNIONE, moduły wysokopoziomowe nie są zależne od tych niższego poziomu.

GRASP

Niektóre z zasad GRASP są niemożliwe do przeanalizowania w kontekście naszego projektu ze względu na jego niewielkie rozmiary.

- **Information Expert (Ekspert):** SPEŁNIONE, informacje są odpowiednio przydzielone do klas.

- **Creator (Twórca):** z uwagi na wielkość projektu, nie potrzebujemy zastosować takiego wzorca.

- **Controller (Kontroler):** z uwagi na wielkość projektu, nie potrzebujemy zastosować takiego wzorca.

- **Low Coupling (Niskie Sprzężenie):** nie jesteśmy w stanie określić, z uwagi na niewielką ilość klas potrzebnych w projekcie.

- **High Cohesion (Wysoka Spójność):** SPEŁNIONE, klasy są skoncentrowane na konkretnych, ściśle określonych zadaniach.

- **Polymorphism (Polimorfizm):** SPEŁNIONE, występuje różnicowanie zachowań operacji w klasach dzięki polimorficznym wywołaniom operacji w klasach pochodnych.

- **Pure Fabrication (Czysty Wymysł):** z uwagi na wielkość projektu, nie potrzebujemy zastosować takiego wzorca.

- **Indirection (Pośrednictwo):** NIESPEŁNIONE, występują bezpośrednie zależności między obiektami.

- **Protected Variations (Ochrona Zmienności):** z uwagi na wielkość projektu, nie potrzebujemy zastosować takiego wzorca.