SOLID принципи

1) Single responsibility - 1 компонент има точно 1 отговорност
   - cohesion - свързаност на компонентите в класа
     Всички компоненти трябва да имат strong cohesion помежду си. Въ противен случай трябва да разделим класа

```
class Triangle {

  getArea();  } strong
  getPer();   } cohesion

  draw();     } strong
  rotate();   } cohesion

};
```

weak cohesion ⟹

```
class Triangle {

  getArea();
  getPer();

};

class TriangleUI {

  draw();
  rotate();

};
```

2) Open-closed principle - всеки клас трябва да бъде отворен за надграждане, но затворен за модификация

3) Liskov substitution principle - трябва да използваме указатели и референции от базов клас, без да се интересуваме кой от наследниците са наследени

4) Interface segregation - потребителите не трябва да разчитат на интерфейс, който не използват

5) Dependency inversion - модулите от по-високо ниво на абстракция трябва да зависят от модули от по-ниско ниво

- модулите от по-високо ниво нямат достъп до данните на тези от по-ниско, а тези от по-ниско работят с базата данни
- няколко зависимост от външни ресурси