```
Public abstract class BaseDecorator implements Rentable

BaseDecorator(Cottage cottage)
        this.cottage = cottage;

@Override
costEquals()
        return cottage.cost;

Public class TVDecorator extend BaseDecorator

TVDecorator(Cottage cottage)
        this.cottage = cottage

@Override
costEquals()
        return super.costEquals() + cost

Public class WinterCalculator
Double coefficient

Public double calculatePrice(double BasePrice)
        return basePrice * coefficient;

Public class Discounter

Public double applyDiscount(double BasePrice)
        return BasePrice * discount;
```

W pracy użyto:

- wzorzec Builder - pozwala na dynamiczne konstruowanie rezerwacji - moglibyśmy na przykład w przyszłości dodać opcje wyjazdu tylko dla dorosłych gdzie setKids nie byłoby używane
- Wzorzec factory - ze względu na istnienie dwóch przypadków domku - base i premium
- Wzorzec decorator - w łatwy sposób pozwala na aktualizowanie ceny po dodaniu danego wyposażenia bądź pakietu dodatkowo pozwalając na łączenie tych udogodnień
- Wzorzec Strategy - pozwala na dwa sposoby obliczania ostatecznej ceny w zależności od momentu sezonu.
- Wzorzec singleton - zapewnia ze w aplikacji istnieje tylko jeden Moderator, który może tworzyć nowe typy domków oraz regulować wspolcznynniki i obniżki.

Kajetan Plewa