Decorator Pattern Aufgabe

1. Implementieren Sie eine `Sandwich` Klassenhierarchie. Abb. 1 zeigt dabei eine unvollständige Klassenhierarchie. Führen Sie alle notwendigen Änderungen durch, so dass das Decorator Pattern angewandt wird. Jeder Sandwich kann mit unterschiedlichen Zutaten belegt werden.

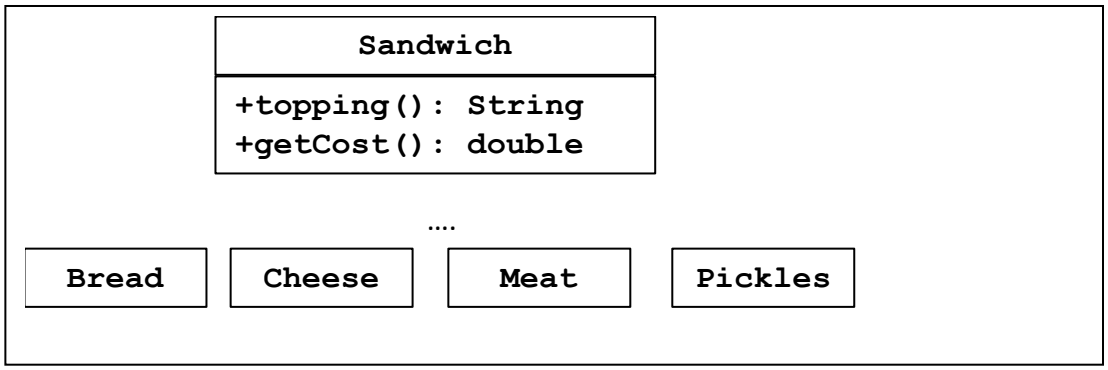

Abb. 1: Sandwich Klassenhierarchie unvollständig

Dabei bedeuten:

- `String topping()`
  gibt den momentanen Belag als String in der Form „Bread with Cheese with Meat…“ etc. zurück
- `double getCost()`
  gibt die Gesamtkosten des gesamten Brötchen inklusive der Beläge zurück

Die Preise der (einzelnen) Zutaten lauten:

- Bread: 0,50 €
- Cheese: 0,50 €
- Meat: 0,60 €
- Pickles: 0,10 €

Eine mögliche Ausgabe verschiedener Sandwich mit unterschiedlichen Belägen sieht beispielsweise so aus:

```
Bread Price: 0.5 €
Bread with Cheese Price: 1.0 €
Bread with Cheese with Meat Price: 1.6 €
Bread with Cheese with Meat with Pickles Price: 1.70 €
Bread with Cheese with Meat with Pickles with Cheese Price: 2.20 €
```

2. Erweitern Sie die Klassenhierarchie um eine weitere Zutat/Belag und belegen einen Sandwich mit Zutaten und der *neuen* Zutat.

3. Erklären Sie in eigenen Worten, wieso das Decorator Pattern hilft, unübersichtliche Vererbungshierarchien zu vermeiden..