1 Java-Tutorial for ISSS-Students - Exercise

A Delivery Service that provides pizzas, sandwiches and drinks wants your help with organising the orders they receive in a program. Implement all the required classes and choose meaningful variables and methods!

1.1 Part 1: Food and Drinks

Create the classes *Pizza*, *Sandwich* and *Beverage* as follows:

- Every Pizza and every Sandwich has a name (e.g. Pizza Diavolo) and a list of ingredients (e.g. salami, red pepper, cheese)
- Every pizza has a diameter (in cm)
- Every sandwich has a length (in cm)
- Every beverage has a name and a capacity (in ml)
- The classes *Pizza* and *Sandwich* extend a common superclass which implements the commonalities of these two classes

1.2 Part 2: Orders

Every order should contain a combination of pizzas, sandwiches and beverages:

- Create Interface *FoodItem* with the methods *getName()* and *getDetails()*
- Let the classes *Pizza*, *Sandwich* and *Beverage* implement the Interface *FoodItem*
- The method *getDetails()* returns a *String* that contains all the relevant information about a single food item (e.g. the ingredients and the diameter of a pizza)
- Create the class *Order* which organises a list of food items

1.3 Part 3: Menu

Create the class *Menu* as follows:

- Every menu contains a list of food items
- The constructor *public Menu()* creates a new menu
- The method *public void addFoodItem(FoodItem item)* adds a given food item to the list of food items of this menu
- The method *public void listMenu()* prints a numbered list of all food items of this menu on the console - this is how it should look like:

1. *Pizza Diavolo (36 cm) – Salami, Red pepper, Cheese*
2. *Pizza Vegetarian (32 cm) – Paprika, Mushroom, Aubergine*
3. *Sandwich Caprese Bacon (24 cm) – Tomato, Mozzarella, Bacon*
4. *Sandwich Tuna (20 cm) – Tuna, Onion, Egg*
5. *Coke (500 ml)*
6. *Lemonade (330 ml)*
7. *...*

- The method *public FoodItem chooseFoodItem()* reads a user input (a number) from the keyboard and returns the *FoodItem* with the given number or *null* if the user input was *0*
  - Make sure that the program doesn't crash due to invalid user input
- The method *public Order order()* displays the menu to the user and asks the user to select food items by number until the user input is *0* - afterwards the complete order of all food items the user selected is returned
  - Tip: use the methods *listMenu* and *chooseFoodItem* that you already implemented

1.4 Part 4: Main

Create the class *Main* with the *main*-method to test your implementation:

- Create a new *Menu* with at least two food items of every class (*Pizza*, *Sandwich* and *Beverage*)
- Take an order from a customer (method *order()*) and print the order on the console