Pflichtaufgabe 29.11.2014

1. Aufgabe: Englisch üben

- Nutze die Englisch Lernplattform mind. 45 min.

2. Aufgabe: Erweitere das Bankensystem

- Mach eine Kopie deines BankenSystem Codes (falls ihr keinen habt stellen wir euch einen)

Klasse BankAccount

- Mach die Klasse BankAccount **abstrakt** und implementiere **zwei unter Klassen**

- Die abstrakte klasse sollte folgende Variablen enthalten: ownerName (String), balance (double), neue Variable hinzufügen: **AccountNumber (String)**

- Die abstrakte Klasse sollte folgende Methoden enthalten: deposit(), withdraw(), getBalance(), neue Methoden hinzufügen: **getAccountNumber(), applyFees()**,
- *applyFees() wird erst in den Unterklassen näher definiert, daher bleibt diese in der abstrakten Klasse leer.*

- Unterklasse 1: **girokonto** erbt von BankAccount
    - applyFees() soll hier eine Kontoführungsgebür berechnen (-5€ pro deposit() oder withdraw() auf den Gesamtwert des Kontos)
    - Ein Girokonto darf bis zu **-2000 Euro ins Minus** gehen dazu muss die Methode withdraw() überschrieben werden

- Unterklasse 2: **savingsAccount** erbt von BankAccount
    - applyFees() soll hier Zinsen berechnen (+1,5% pro deposit() oder withdraw() auf den Gesamtwert des Kontos.
    - *Herausforderung(freiwillig): es dürfen nur 3 aktionen pro Login Session getätigt werden*

Klasse Bank

- Wenn ein **neuer Account** angelegt wird soll **abgefragt** werden ob es ein **Savings** oder ein **girokonto** sein soll
- Die Klasse Bank soll um einen **Kontonummergenerator:** accountNumbergenerator() erweitert werden
- Eine Kontonummer (String) soll 6 Zeichen lang sein
- Die Kontonummer soll eine Zufallszahl sein, du hast zur Auswahl die **Math Class** oder die **Random Class**, schreibe in einem kurzen **Kommentar warum** du dich für eine der beiden entschieden hast?
    - Die Nummer (int) musst du noch für die Rückgabe in einen String **Casten**
    - Eine Kontonummer darf nur **einmal existieren**

- **findAccount()** soll jetzt anhand der **Kontonummer** den Account suchen

- Führt einen weiteren **Switch-Case** ein bei dem man sich mit Name und accountNumber “anmeldet” und dann die daten im System “gespeichert” sind, damit man die accountNumber nicht bei jeder Transaktion erneut eingeben muss.

- *Programmstart:*
  Case 1: neuen Account erstellen
  Case 2: Login

  Wenn angemeldet:
  Case 1: Deposit
  Case 2: Withdraw
  Case 3: Check Balance
  Case 4: beenden

3. Aufgabe: freiwillige Herausforderung: Kontostand in Binärzahl ausgeben

- Erstellt eine weitere Methode **decimalToBinary()**
  Dieser übergebt ihr den Kontostand und **castet** diesen in ein **Int** und berechnet dann die Binärdarstellung.
- Dafür wird eine Rekursion benötigt
- Die Zahl wird durch 2 geteilt und der Rest wieder mit 2 geteilt usw. der jeweilige Rest wird jeweils mit **%2** ausgegeben.