Assignment 29 – SavingsAccount

Doel

Het doel van deze opdracht is het oefenen met subclasses en superclasses, met het juiste gebruik van access modifiers.

Overzicht

Deze opdracht heeft 3 classes:

- Class Assignment29
- Class Account (eerder aangemaakt)
- Class SavingsAccount

1. Voorbereiding

a. Maak een package aan met de naam **assignment29** (let op packages beginnen altijd met een kleine letter).

b. Maak in de package een nieuwe class met de naam **Assignment29** en maak de **main** methode aan. De verdere implementatie van de **main** methode wordt verder op in de opdracht besproken.

2. Class Account

In de vorige opdracht hebben we de (super)class **Account** aangemaakt. Deze zal worden gebruikt door onze nieuw aan te maken class **SavingsAccount**.

a. Voeg 2 methoden toe aan deze class:
   i. withdraw() – om geld op te nemen, met als input parameter de hoeveelheid om op te nemen.
   ii. deposit() – om geld te storten, met als input parameter de hoeveelheid om te storten.

3. Class SavingsAccount

Vervolgens maken we de (sub)class **SavingsAccount**, die de eigenschappen en methoden erft van **Account**.

a. Maak in de package assignment29 een nieuwe class met de naam **SavingsAccount**. Deze class erft de eigenschappen en methoden van **Account**.

b. Voeg een field toe aan de class **SavingsAccount** met de naam **interest**. Het leuke van een spaarrekening is dat je rente ontvangt (wie het kleine niet eert..).

   Bepaal zelf meest voor de hand liggend data type.

c. Maak een nieuwe methode in de class **SavingsAccount** met de naam **getAnnualInterest** die de jaarlijkse rente berekent op basis van **balance** en **interest** en deze waarde teruggeeft.

d. De class SavingsAccount heeft een constructor waarbij je geen input parameter kan meegeven. De constructor zorgt ervoor dat de volgende waarden zijn ingesteld:

    a. name = SavingsAccount
    b. balance = 0
    c. interest = 5

e. Welke access modifiers kan je nu gebruiken in de class **Account** voor de fields **name** en balance?

    {access modifier} {data type} **name**;
    {access modifier} {data type} **balance**;

f. Welke access modifier van de werkende access modifiers vind jij het beste en waarom?

4. Class Assignment29

De **main** methode hebben we al aangemaakt in de Voorbereiding (1b). Nu gaan we de implementatie doen van de **main** methode.

a. Instantieer van de class **SavingsAccount**, een nieuw object met de naam **raboSpaarrekening**.

b. Zorg ervoor dat er geld op komt te staan (5000) en laat zien wat de jaarlijkse rente is.

c. Roep de methode **printOverview()** aan van het aangemaakte object **raboSpaarrekening**. Welke output krijg je?

d. Haal er geld af (30000) en roep de methode **printOverview()** aan.

e. Gebruik ***override*** om een 'nieuwe' methode **withdraw()** te maken voor de class **SavingsAccount**. We willen niet dat mensen een negatief saldo kunnen hebben op hun spaarrekening. Los dit probleem op en laat evt een correcte foutmelding zien.

f. Instantieer van de class **SavingsAccount**, een nieuw object met de naam **abnSpaarrekening**. Zet hier 1000 op en probeer er 2000 van af te halen.