OOP – Objektorientierte Programmierung

Übung: 6.1 Labyrinth

In den Teilaufgaben a und b sollen Sie BlueJ interativ benutzen!

a. **Schatzsuche**
Führen Sie den roten Schatzsucher zum goldenen Schatz.
Rufen Sie durch Rechtsklick auf dem orangen Klassensymbol **`Schatzsucher`** das Kontextmenü auf. Erzeugen Sie von dieser Klasse ein Objekt, in dem Sie dort den Konstruktor mit dem Befehl **`new Schatzsucher()`** aufrufen und geben Sie dem neuen Objekt einen Namen z. B. **`jackSparrow`**. Durch Rechtsklick auf dem roten Objektsymbol sehen Sie eine Übersicht der vorhandenen Methoden. Rufen Sie die Methode **`geheNachRechts()`** auf. Probieren Sie danach die Methode **`geheNachRechts(int strecke)`** auf und übergebne Sie dem Parameter **`strecke`** vom Datentype **`int`** das Argument 4.
Führen Sie die Schatzsuche fort, bis der Schatzsucher sein Zeile erreicht hat!

b. **Unsichtbarerer Geist**
Die Geisterstunde ist vorbei und der unsichtbare Geist muss wieder zurück in seine Truhe
Erzeugen Sie ein Objekt der Klasse **`UnsichtbarerGeist`** und geben sie ihm den Namen **`hugo`**! Rufen sie nun die Bewegungsmethoden in BlueJ interaktiv auf bis der Geist in seiner Truhe ist. Tipp: Durch Doppelklick auf dem roten Objekt-Symbol öffnet sich der Objekt-Inspektor und Sie können feststellen, wo sich der Geist befindet.

Ab jetzt sollen die Objekte und Methoden nicht mehr interaktiv steuern, sondern die Aufgaben mit Java-Befehlen lösen. Öffnen Sie dazu die Klasse **`Aufgaben`** und vervollständigen Sie dort die Methoden **`aufgabeA`** bis **`aufgabeM`**, so dass sie die folgenden Teilaufgaben lösen!

a. **Schatzsuche (als Programm)**
Führen Sie den roten Schatzsucher zum goldenen Schatz.
Rufen Sie durch Rechtsklick auf dem orangen Klassensymbol **`Aufgaben`** das Kontextmenü auf und rufen Sie die Methode **`void aufgabeA()`** aus! Versuchen Sie den vorhanden Java-Code der Methode zusammen mit den Kommentaren zu verstehen und vervollständigen die Methode bis Schatzsucher den Schatz Programm-gesteuert in nur 37 Schritten erreicht. Sie können nach jedem Programmierschritt die Klasse **`Aufgaben`** übersetzen und überprüfen, wie weit sich der Schatzsucher schon seinem Ziel genähert hat.

b. **Unsichtbarerer Geist (als Programm)**
Die Geisterstunde ist vorbei und der unsichtbare Geist muss wieder zurück in seine Truhe.
Schreiben Sie nun ein Programm (Methode **`aufgabeB()`** in der Klasse **`Aufgaben`**), das nun auch die Teilaufgabe B Programm-gesteuert löst! Der Beispiel-Code zur Aufgabe C auf der folgenden Seite kann Ihnen eventuell dabei helfen.

c. **Labyrinth**

Suchen Sie den kürzesten Pfad durch das Labyrinth. Erzeugen sie ein Objekt der Klasse **`Roboter`** und führen sie den hellgrünen Roboter zu seinem dunkelgrünen Ziel. Deklarieren Sie eine Variable **`robo1`** vom Datentyp der Klasse **`Roboter`**. Erzeugen Sie durch Konstruktoraufruf ein Roboter-Objekt und weisen Sie es der Variablen **`robo1`** zu.

Das folgendes Beispiel-Java-Programm kann Ihnen dabei helfen. Wenn Sie die Klasse übersetzen, können Sie die Methode aufrufen und die programmierten Befehle werden ausgeführt. Vervollständigen Sie die Methode **`aufgabeC()`** bis die Aufgabe C vollständig gelöst ist.

```
public class Aufgaben
{
      public static void aufgabeC()
      {
            // Deklaration der lokalen Variablen robo1
            RoboterGruen robo1;

            // Erzeugung eines neuen Objektes der Klasse Roboter
           // durch Konstruktoraufruf
            // und Zuweisung an die Variable robo1
            robo1 = new RoboterGruen();

            // Aufruf der Methode geheNachUnten
            // des Objektes robo1 ohne Argument
            robo1.geheNachUnten();

            // Aufruf der Methode geheNachUnten mit Argument
            robo1.geheNachUnten( 2 );
      }
}
```

d. **Schaf und Wolf**

Schaf und Wolf sollen ihren Platz tauschen, ohne dass der Wolf das Schaf fressen kann.

Erzeugen sie zwei Objekte der Klassen Schaf und Wolf und bewegen Sie durch manuelle Methodenaufrufe die beiden Objekte auf ihre Zielplätze.

e. **Sokoban-Rätsel**

Bei dem bekannten Sokoban-Rätseln muss ein Packetschieber (grüner Punkt) alle Pakete (hell-gelbe Punkte) an ihre Zielplätze (dunkel-gelbe Punkte) schieben. Leider können die Pakete nur geschoben und nicht gezogen werden.

Als Vorbereitung kannst du den Sokoban auch mit den Pfeiltasten bewegen. Zur Lösung der Programmieraufgabe musst du das Objekt durch Methodenaufrufe steuern, die du in die Methode **`aufgabeE`** bis **`aufgabeM`** schreibst.

Wenn du als erster fertig bist, gib den Lehrern Bescheid!

Viel Erfolg!