Blatt 7

Niels Haupt: 467970

Ughur Alakbarov: 436904

Aufgabe 3

a)
1. Konstruktoren:
    1. `A(int)` – 100 ist ein `int`
    2. `A(String)` – "written in A(int)" ist ein `String` `print`-Anweisungen:
    3. "v1.x: written in A(int)"
2. Konstuktoren:
    1. `B(int)` – 100 ist ein `int`
    2. `B(String)` – "written in B(String)" ist ein `String`
    3. `A(String)` – "written in B(String)" ist ein `String` und die Oberklasse von `B` ist `A`. `print`-Anweisungen:
    4. "v2.x: written in B(String)" – `v2` hat den Deklarationstyp `A` und `v2.x` greift somit auf `A.x` zu, welches in `A(String)` auf "written in B(String)" gesetzt wurde
    5. Kompilierfehler – `v2` wurde als `A` deklariert und darf somit nicht zu `B` gecastet werden, da `B` eine Unterklasse von `A` ist
3. Konstuktoren:
    1. `B(A)` – `v2` hat den Deklarationstyp `A`
    2. `B(String)` – "written in B(A)" ist ein `String`
    3. `A(String)` – "written in B(String)" ist ein `String` und die Oberklasse von `B` ist `A`. `print`-Anweisungen:
    4. "((A) v3).x: written in B(String)" – `v3` wird gecastet zu `A`; es wird auf `A.x` zugegriffen, welches in `A(String)` auf "written in B(String)" gesetzt wurde
    5. "v3.x: written in B(A)" – `v3` hat den Deklarartionstyp `B`; `v3.x` greift auf `B.x` zu, welches in `B(String)` auf "written in B(A)" gesetzt wurde
4. Konstuktoren:
    1. `B()` – wurde ohne Argumente aufgerufen
    2. `B(String)` – "written in B()" ist ein `String`
    3. `A(String)` – "written in B(String)" ist ein `String` und die Oberklasse von `B` ist `A`. `print`-Anweisungen:
    4. "((A) v4).x: written in B(String)" – `v4` wird gecastet zu `A`; es wird auf `A.x` zugegriffen, welches in `A(String)` auf "written in B(String)" gesetzt wurde
    5. "v4.x: written in B()" – `v4` hat den Deklarartionstyp `B`; `v4.x` greift auf `B.x` zu, welches in `B()` auf "written in B()" gesetzt

wurde

b)
1. `A.f(A)`
2. `A.f(A)` – anhand des Deklarationstyps von `v2`
3. `A.f(A)` – `v3` wird hochgecastet
4. `A.f(A)`
5. `A.f(A)` – anhand des Deklarationstyps von `v2`
6. `A.f(A)` – `v3` wird hochgecastet
7. `B.f(A)`
8. `B.f(A)`
9. `B.f(B)`

Aufgabe 5

```java
public sealed class Pflanze permits BlauerEisenhut, Salbei, Rosengewaechs {

  protected int maxLaenge;
  protected int wachstum;
  protected int laenge;

  public Pflanze(int maxLaenge, int wachstum, int laenge) {
    this.maxLaenge = maxLaenge;
    this.wachstum = wachstum;
    this.laenge = laenge > maxLaenge ? 0 : laenge;
  }

  public final int getMaxLaenge() {
    return this.maxLaenge;
  }

  public final int getWachstum() {
    return this.wachstum;
  }

  public final int getLaenge() {
    return this.laenge;
  }

  public void waessern() {
    this.laenge = Math.min(this.laenge + this.wachstum, this.maxLaenge);
  }

  public void schneiden(int x) {
    if (x < 0) return;
    this.laenge = Math.max(0, this.laenge - x);
```

```
  }
}
```

```
public final class BlauerEisenhut extends Pflanze {

  public BlauerEisenhut() {
    super(Integer.MAX_VALUE, 1, 1);
  }

  @Override
  public void schneiden(int x) {
    if (x < 0) return;
    this.laenge = 1;
  }
}
```

```
public final class Salbei extends Pflanze {

  public Salbei() {
    super(6, 1, 1);
  }

  /*
   * Methods
   */

}
```

```
public sealed class Rosengewaechs extends Pflanze permits Lorbeerkirsche, Himbeere {

  protected int verbreitung;

  public Rosengewaechs(int maxLaenge, int wachstum, int laenge, int verbreitung) {
    super(maxLaenge, wachstum, laenge);
    this.verbreitung = verbreitung;
  }

  public int getVerbreitung() {
    return this.verbreitung;
  }

  @Override
  public void waessern() {
```

```
    int x = this.verbreitung * this.wachstum;
    this.laenge = Math.min(this.laenge + x, this.maxLaenge);
  }

  @Override
  public void schneiden(int x) {
    if (x < 0) return;
    this.laenge = Math.min(this.laenge, 1);
  }
}
```

```
public final class Lorbeerkirsche extends Rosengewaechs {

  public Lorbeerkirsche() {
    super(20, 2, 1, 3);
  }

  @Override
  public void schneiden(int x) {
    if (x < 0) return;
    this.laenge = Math.max(0, this.laenge - (x / 2));
  }
}
```

```
public final class Himbeere extends Rosengewaechs {

  public Himbeere() {
    super(10, 1, 1, 2);
  }

  /*
   * Methods
   */
  @Override
  public void schneiden(int x) {
    if (x < 0) return;
    this.laenge = Math.max(0, this.laenge - x);
  }
}
```

```
public record PflanzenPaar(Pflanze a, Pflanze b) {}
```

```java
public class Feld {

  public static Pflanze auswahl(PflanzenPaar pair) {
    Pflanze p =
        switch (pair) {
          case PflanzenPaar(Rosengewaechs x, _) -> x;
          case PflanzenPaar(_, Rosengewaechs x) -> x;
          case PflanzenPaar(BlauerEisenhut e, Salbei s) when s.getLaenge() >= 5 -> e;
          case PflanzenPaar(Salbei s, BlauerEisenhut e) when s.getLaenge() >= 5 -> e;
          default -> pair.a();
        };

    if (p instanceof Rosengewaechs) {
      while (p.getLaenge() < p.getMaxLaenge()) {
        p.waessern();
      }
      if (p instanceof Himbeere) {
        p.schneiden(1);
      }
    }

    return p;
  }

  public static void main(String[] args) {
    System.out.println("Start");
    Himbeere himbeere = new Himbeere();
    Lorbeerkirsche lorbeerkirsche = new Lorbeerkirsche();
    Salbei salbeiLang = new Salbei();
    lorbeerkirsche.waessern();
    lorbeerkirsche.schneiden(6);
    salbeiLang.waessern();
    salbeiLang.waessern();
    salbeiLang.waessern();
    salbeiLang.waessern();
    Salbei salbei = new Salbei();
    BlauerEisenhut blauerEisenhut = new BlauerEisenhut();
    blauerEisenhut.schneiden(1);

    PflanzenPaar pair1 = new PflanzenPaar(himbeere, salbei);
    PflanzenPaar pair2 = new PflanzenPaar(salbei, lorbeerkirsche);
    PflanzenPaar pair3 = new PflanzenPaar(lorbeerkirsche, blauerEisenhut);
    PflanzenPaar pair4 = new PflanzenPaar(salbeiLang, blauerEisenhut);
    PflanzenPaar pair5 = new PflanzenPaar(salbei, blauerEisenhut);

    System.out.println("Auswahl:" + auswahl(pair1) + ", Laenge:" + auswahl(pair1).getLaenge(
```

```
        System.out.println("Auswahl:" + auswahl(pair2) + ", Laenge:" + auswahl(pair2).getLaenge(
        System.out.println("Auswahl:" + auswahl(pair3) + ", Laenge:" + auswahl(pair3).getLaenge(
        System.out.println("Auswahl:" + auswahl(pair4) + ", Laenge:" + auswahl(pair4).getLaenge(
        System.out.println("Auswahl:" + auswahl(pair5) + ", Laenge:" + auswahl(pair5).getLaenge(
    }
}
```