Musterlösung Probeprüfung Informatik I D-BAUG Sommer 2020

Aufgabe 1: Java Basics

- Lesender Zugriff: Zeilen 10, 17 und 18
- Fehler erkennen: Zeile 18: `result[j++] = array[i]`
- Variablen Scope: Zeilen 9-13
- Werte: `array = [2, 13, 58, 3, 9, 13, 24]`, `result = [2,58,3,9,24]`
- Postcondition: The resulting array is a new instance that contains all elements from the original array and in the same order, except for the elements that are equal to 'value', which are filtered out. That is, all occurrences of 'value' are removed from the original array. No other elements are added.

Aufgabe 2: Vererbung und Polymorphie

```
Car c = new Car(true,"Anna");
c.print();
```

Ausgabe / Output: Anna's car 5 seats

```
Vehicle v = new Bicycle(false,"Bob");
v.print();
```

Ausgabe / Output:Bob's bicycle, keeps fit!

```
Bicycle b = new PrivateVehicle(5,"Celine",true);
Out.println(b.requiresMuscles());
```

Compiler Error: PrivateVehicle is abstract and cannot be instantiated

---

```
Vehicle v = new Car(false,"Dave");
Out.println(v.requiresMuscles());
```

Compiler Error: cannot find symbol requiresMuscle

```
PrivateVehicle pv = new Bicycle(true,"Eva");
Bicycle b = pv;
Out.println(b.electric)
```

Compiler Error: incompatible types: cannot be converted to Bicycle

Aufgabe 3: Dynamische Datenstrukturen

```
public class Queue {
  Node oldest = null; // The oldest element should be the head of the list
  Node newest = null; // The newest element should be the tail of the list

  // POST: 'value' is added to the end of the queue
  public void enqueue(int value){
    Node n = new Node(value, null);
    if (empty()){
      newest = oldest = n;
    } else {
      newest.next = n;
      newest = n;
    }
  }

  // PRE: queue must not be empty
  // POST: the first value in the queue is returned and removed from the queue
  public int dequeue(){
    assert !empty();
    Node toRemove = oldest;
    if (oldest == newest){
      oldest = newest = null;
    } else {
      oldest = oldest.next;
    }
    return toRemove.value;
  }

  // POST: Yields 'true' if the queue is empty and 'false' otherwise.
  public boolean empty(){
    return oldest == null;
  }
}
```

Siehe code-expert