Aufgabe 1 (45 Punkte)

Parkable (2.5 Punkte)

```
public interface Parkable { // 0.5
  public int getHeight(); // 0.5
  public int getWidth(); // 0.5
  public String getName(); // 0.5
  public double getValue(); // 0.5
}
```

ParkingLot (8.5 Punkte)

```
public class ParkingLot { // 0.5
  public static final int MIN_WIDTH = 200; // 0.25
  public final int number; // 0.25
  public final int width; // 0.25
  private Parkable item; // 0.25

  public ParkingLot(int number, int width) { // 0.5
    this.number = number; // 0.25
    this.width = width; // 0.25
  }

  public void park(Parkable item) throws TooWideException { // 1
    if (item.getWidth() > width) { // 0.5
      throw new TooWideException(item.getWidth() - width); // 1
    } else {
      this.item = item; // 0.5
    }
  }

  public boolean isFree() { // 0.5
    return item == null; // 0.5
  }

  public double getValue() { // 0.5
    if (isFree()) { // 0.5
      return 0; // 0.5
    } else {
      return this.item.getValue(); // 0.5
    }
  }
}
```

TooWideException (2 Punkte)

```
public class TooWideException extends Exception { // 0.5
  public final int difference; // 0.25

  public TooWideException(int difference) { // 0.5
    super(); // 0.5
    this.difference = difference; // 0.25
  }
}
```

Floor (13 Punkte)

```
public class Floor implements Comparable<Floor> { // 0.5
  public static final int MIN_HEIGHT = 200; // 0.25
  private int height; // 0.25
  private int level; // 0.25
  public final ArrayList<ParkingLot> parkingLots; // 0.25

  public Floor(int level, int height) { // 0.5
    this(level, height, 2); // 0.5
  }

  public Floor(int level, int height, int numberOfParkingLots) { // 0.5
    this.level = level; // 0.25
    this.height = height; // 0.25
    this.parkingLots = new ArrayList<>(); // 0.5
    int width = ParkingLot.MIN_WIDTH; // 0.5
    for (int i = 0; i < numberOfParkingLots; i++) { // 0.5
      this.parkingLots.add(new ParkingLot(i + 1, width)); // 0.5
      width = width + 10; // 0.5
    }
  }

  public boolean park(Parkable item) { // 0.5
    if (item.getHeight() >= this.height) { // 0.5
      return false; // 0.5
    } else {
      for (ParkingLot parkingLot : parkingLots) { // 0.5
        if (parkingLot.isFree()) { // 0.5
          try { // 0.25
            parkingLot.park(item); // 0.5
            System.out.println(
                item.getName()
                    + " geparkt. Stock: "
                    + level
                    + " Platz: "
                    + parkingLot.number); // 0.5
            return true; // 0.25
          } catch (TooWideException e) { // 0.5
            System.out.println(
                "Parkplatz um "
                    + e.difference
                    + " zu klein. Stock: "
                    + level
                    + " Platz: "
                    + parkingLot.number); // 0.5
          }
        }
      }
      return false; // 0.5
    }
  }

  public int compareTo(Floor other) { // 0.5
```

```
    if (other.level > this.level) { // 0.5
      return -1; // 0.25
    } else {
      return 1; // 0.25
    }
  }
}
```

FloorValueComparator (6 Punkte)

```
public class FloorValueComparator implements Comparator<Floor> { // 0.5

  public int compare(Floor floor1, Floor floor2) { // 0.5
    double v1 = getFloorValue(floor1); // 0.5
    double v2 = getFloorValue(floor2); // 0.5
    if (v1 > v2) { // 0.5
      return -1; // 0.25
    } else if (v2 > v1) { // 0.5
      return 1; // 0.25
    } else {
      return 0; // 0.25
    }
  }

  private double getFloorValue(Floor floor) { // 0.5
    double value = 0; // 0.25
    for (ParkingLot parkingLot : floor.parkingLots) { // 0.5
      value += parkingLot.getValue(); // 0.5
    }
    return value; // 0.5
  }
}
```

CarPark (7 Punkte)

```
public class CarPark { // 0.5
  public final ArrayList<Floor> floors; // 0.25

  public CarPark(int numberOfFloors) { // 0.5
    this.floors = new ArrayList<>(); // 0.25
    int height = Floor.MIN_HEIGHT; // 0.25
    for (int i = 0; i < numberOfFloors; i++) { // 0.5
      this.floors.add(new Floor(i, height)); // 0.5
      height = height + 25; // 0.25
    }
  }

  public void park(Parkable item) { // 0.5
    for (Floor floor : floors) { // 0.5
      if (floor.park(item)) { // 0.5
        break; // 0.5
      }
    }
  }
```

```
  public void sortByValue() { // 0.5
    Collections.sort(this.floors, new FloorValueComparator()); // 0.5
  }

  public void sortByDefault() { // 0.5
    Collections.sort(this.floors); // 0.5
  }
}
```

ExamTask01 (6 Punkte)

```
public class ExamTask01 { // 0.5
  public static void main(String[] args) { // 0.5
    CarPark carPark = new CarPark(5); // 0.5
    carPark.park(new BMW(100, 100, "BMW 320i", 30_000)); // 0.5
    carPark.park(new BMW(250, 200, "BMW 340i", 60_000)); // 0.5
    carPark.sortByValue(); // 0.5
    Floor firstFloor = carPark.floors.get(0); // 0.5
    double value = 0; // 0.5
    for (ParkingLot parkingLot : firstFloor.parkingLots) { // 0.5
      if (parkingLot.getValue() > value) { // 0.5
        value = parkingLot.getValue(); // 0.5
      }
    }
    System.out.println("Wert des wertvollsten Autos im teuersten Stock: " + value); // 0.5
  }
}
```