

Lab Test 2, Winter 2025
Course: COSC 3P91

Number of Pages: 2
Number of Hours: 2
Instructor: M. Winter

Instructions

- 1) Write a program as indicated below using Java and NetBeans.
- 2) Zip your project folder and name it with your name and student number, e.g., MichaelWinter123456.zip
- 3) Send the file to your lab instructor by email.
- 4) Wait for a confirmation email.
- 5) Log out of the lab computer and hand back the question sheets to the lab instructor. You can now leave the lab.
- 6) The test is worth a total of 20 marks.

In this test you are going to use maps and the design patterns **Strategy** and **Singleton**. Implement the following:

1. (1 mark) A class `Car` whose object have a `private final` field called `description` (type `String`). In addition to a constructor that receives the description, the class must provide an appropriate implementation of the `toString()` method.
2. (3 marks) A class `MyCar` that is a `Car` and has description "This is my car". There can only be one instance of `MyCar` at any time, i.e., `MyCar` must implement the **Singleton** design pattern.
3. (8 marks) A class `Person` with the following features:
 - a. A person has three private fields of a name (type `String`), age (type `int`), and occupation (type `String`). The constructor of the class receives values for all three fields.
 - b. Provide access methods for the fields. All fields can be read, and age and occupation can be set.
 - c. Implement the method `toString()` appropriately.
 - d. Make the class `Comparable`. A person `p1` is "smaller" than another person `p2` if the `p1`'s name is smaller in the lexicographic order than `p2`'s name (`compareTo` on `String`).
4. (1 mark) A class `AgeComparator` that is a `Comparator` and compares to persons by their age.
5. (7 marks) A class `OwnershipTable` that is a map between `Person` and `Car`. In addition to the methods of a map, the class must have the following features:

Brock University

- a. Two methods `ownersByName()` and `ownersByAge()`. Both methods return a list of persons that is sorted by name (default order on `Person`) or by age, respectively. Use the `Collections.sort` method.
- b. A method `ownersOf(Car car)` that returns a `Set<Person>` of all persons that own the car `car`. Comparison between cars must be done by `==`, i.e., identical objects.

Reminder: A class that is `Comparable` must also implement the method `equals(Object obj)` appropriately. If a class is used as key values of a map, the class must always implement the methods `equals(Object obj)` and `hashCode()` appropriately.