Université Grenoble Alpes 2024-2025
M1 AM

Object-oriented and software design - C++
Test 1

Duration : 1h30 - No documents allowed
Only use the notions of C++ presented so far

NAME…………………………………..……………. FIRST NAME………………………………………..

The goal in this exercise is to define classes that will be useful to *manage the sale of train tickets*.

First, we will consider the definition of a class to model *locations*. To that goal, class `Location` will characterize a location, as **a *city* and a *country*** (both as `char *`). This class has a method **`same_country`** that compares the country with the country of another Location `loc`, it returns `true` if those countries are identical, `false` otherwise.

Question 1. Give the **declaration** of class `Location`, **carefully explain** the signature and the role of each method.

Answer. 2,5 pts

Question 2. Give the **definitions** of all the methods of class `Location`.

Answer. 4,5 pts

Question 3. Define an overloading of **operator <<** for this class `Location`.

Answer. 1 pt

Do you **need to modify** class `Location` to do so (yes and how / no)? Explain why.

Answer. 1 pt

We now assume the existence of the following class `Time` to model a time, with the attributes hour and minutes:

```
class Time {
private :
  int hour, minutes;
public :
  Time(int h, int m) {
    hour = h;
    minutes = m;
  }
  string get_time() const {
    string s;
    s.append(to_string(hour)).append(":").append(to_string(minutes));
    return s;
  }
};
```

Question 4. Define a class `Train_ticket` to characterize train tickets, represented by:
- a ***departure location***, an ***arrival location***,
- a (statically allocated) **C-style array of 2 instances of** `Time`, for the departure time and the arrival time,
- a ***raw price*** (float), and a ***reduction rate*** (float).

When train tickets are created, the reduction rate is unknown (assumed 0) - only the **departure city**, the **departure country**, the **arrival city,** the **arrival country**, the hour and minutes of the **departure time**, the hour and minutes of the **arrival time**, and the **raw price** are provided.
This class has a **setter** for the reduction rate (for example, the reduction rate equals 20 if the reduction percentage is 20%), a method **`ticket_price`** that computes the actual price by

applying the reduction rate to the raw price, and a method **`international_train`** that returns `true` if the train is international, `false` otherwise.

Answer. 5 pts

Do you **need to modify** class `Location` and/or class `Time` do so? If yes, explain what and why ; if no, explain why.

Answer. 1,5 pt

Question 5. Define a function `main` that declares a **train ticket** from Grenoble, France to Lyon, France, with a raw price of 20.4 euros, departure time 10:20 and arrival time 11:40, and a **train ticket** from Paris, France to Brussels, Begium, with a raw price of 203.8 euros, departure time 12:32 and arrival time 15:10, and finally **sets** the reduction rate of the second ticket to 10.

Answer. 1 pt

Question 6. Define on overloading of **operator <<** for class `Train_ticket`: it inserts on the output stream the **departure** location and time, the **arrival** location and time, the actual **ticket price** and a specific message that indicates whether the train is **international or not**.

Answer. 2 pts

Do you need to **perform modifications** in the answers to the previous questions to do so? If yes, explain what and why ; if no, explain why.

Answer. 1,5 pt

Appendix.

```
NAME
     strcmp -- compare strings

SYNOPSIS
     #include <string.h>

     int strcmp(const char *s1, const char *s2);

DESCRIPTION
     The strcmp() function compares the two strings s1 and s2. It returns an integer
     less than, equal to, or greater than zero if s1 is found, respectively, to be
     less than, to match, or be greater than s2.
...
```