Object Oriented Programming Lab

Lab 03 **Marks 20**

Instructions

- Work in this lab individually.
- You can use your books, notes, handouts etc. but you are not allowed to borrow anything from your peer student.
- Make sure to follow the best coding practices.
- Include comments to explain the logic where necessary.
- *You are strictly **NOT ALLOWED** to include any additional data-members/functions/constructors in your class.*
- Test your program thoroughly with various inputs to ensure proper functionality and error handling.
- Show your work to the instructor before leaving the lab to get some or full credit.

ADT: Time

Write a class named **Time** that represents time in a 24-hour format, with the following functionalities:

1. The class should have the following three private data members:

- An integer named **second** that holds the value of the seconds.
- An integer named **minute** that holds the value of the minutes.
- An integer named **hour** that holds the value of the hours.

Ensure that values assigned to **second**, **minute**, and **hour** are within their respective valid ranges:

- **second**: 0 to 59 (inclusive)
- **minute**: 0 to 59 (inclusive)
- **hour**: 0 to 23 (inclusive)

2. Implement mutators to set the values of **second**, **minute**, and **hour**, and accessors to retrieve their values.

3. Provide constructors:

- A constructor accepting **second**, **minute**, and **hour** as arguments. These values should be assigned to the object's appropriate member variables.
- A constructor accepting **minute** and **hour** as arguments. The second member should be assigned the default value (0).
- A default constructor that initializes all data members of the class with default values.

4. Implement the following overloaded operators:

- **Stream insertion (<<)** to display the time in the format **16:50:45** (hour:minutes:seconds).
- **Stream extraction (>>)** to prompt the user for a time input in the format **hour:minutes:seconds**.
- **Pre-increment (++)** and **pre-decrement (--)** to increment and decrement the **second** member of the object, respectively.
- **Post-increment (++)** and **post-decrement (--)** to increment and decrement the **second** member of the object, respectively.
- **Binary subtraction (-)** to subtract one time from another and return the number of seconds between two times. For example, if **16:50:45** is subtracted from **16:51:00**, the result will be **15**.
- **Unary Addition (+)** to return **true** if the time is within working hours (09:00:00 to 17:00:00), **false** otherwise.

5. The class should handle the following conditions:

- When a time is set to the last second of the minute and incremented, it should become the first second of the following minute.
- When a time is set to **59:59** (minute:second) and incremented, it should become **00:00** (minute:second) of the following hour.
- When a second is set to the first second of the minute and decremented, it should become the last second of the previous minute.
- When a time is set to **00:00** (minute:second) and decremented, it should become **59:59** (minute:second) of the previous hour.

6. In the **main** function, create instances of the **Time** class and demonstrate the functionality of each function clearly.