**National University of Computer and Emerging Sciences**



**Object Oriented Programming**

**Lab Manual 7**

|  |  |
| --- | --- |
| Course Instructor | Hafiz M. Hamza |
| Lab Instructor(s) | Mian Basam  Shakeel Zafar |
| Section | F |
| Semester | Spring 2020 |

Department of Computer Science

FAST-NU, Lahore, Pakistan

**Task** 1

Implement a class named *DateTime* whose objects store and manipulate date and time.

**class** *DateTime*{

**private**:

**unsigned int** *month*; **unsigned int** *year*; **unsigned int** *day*; **unsigned int** *hour*; **unsigned int***minute*; **unsigned int***second*;  **unsigned int** *millisecond*;

**public**:

// ... };

**Task 1.1**

Implement a function named *isLeapYear* which takes a *year* value as input, and returns **true**/**false** on whether the *year* is leap year or not. Every year that is exactly divisible by 4 is a leap year, except for years that are exactly divisible by 100, but these centurial years are leap years if they are exactly divisible by 400. For example, the years 1700, 1800, and 1900 were not leap years, but the years 1600 and 2000 were.

**static bool** *isLeapYear*(**unsigned int** *year*);

**Task** **1.2**

Implement a function named *addYears* which takes a value and adds it in *year* data member. *addMonths* should add the *months* value to *month* data member. If the value of *month* goes above the allowed range of month (1 - 12), for example, if the previous value was 7, and the user asks to add 10 months in the date, the new *month* value will become 17, which is obviously not in the allowed range, is should make it to be in the actual range, and add years instead, like here, *month* will become 5, and 1 will be added to *year*. Use *addYears* method to add years.

* **void** *addYears*(**unsigned int** *years*);
* **void** *addMonths*(**unsigned int** *months*);

**Task** **1.3**

Implement a function named *addDays* which takes a value and adds it in *day* data member. If the value of *day* goes above the allowed range of month, then it should make the *day* value to be in the allowed range and add months instead. Use *addMonths* method to add months.

1. January – (1 – 31)
2. February (1 – 28 if the year is not leap year, otherwise 29)
3. March – (1 – 31)
4. April – (1 – 30)
5. May – (1 – 31)
6. June – (1 – 30)
7. July – (1 – 31)
8. August – (1 – 31)
9. September – (1 – 30)
10. October – (1 – 31)
11. November – (1 – 30)
12. December – (1 – 31)

**Task 1.4**

For range, you should implement a function named *getLastDayOfMonth*, which should return the last day of current month based on above criteria. Now, the range will be 1 – *getLastDayOfMonth*().

* **unsigned int** *getLastDayOfMonth*()
* **const**; **void** *addDays*(**unsigned int** *days*);

**Task1.5**

Implement the functions *addHours*, *addMinutes*, *addSeconds*, and *addMilliseconds* which takes values for *hours*, *minutes*, *seconds*, and *milliseconds* and add them in *hour*, *minute*, *second*, and *millisecond* data members respectively. If the data member values go above the allowed range, it should make it to be in the allowed range, and add the bigger values (like if *minute* go above the range, add hours by calling *addHours* and so on). The allowed range for *hour* is (0 – 23), for *minute* is (0 – 59), for *second* is (0 – 59), and for *millisecond* is (0 – 999).

* **void** *addHours*(**unsigned int** *hours*);
* **void** *addMinutes*(**unsigned int** *minutes*);
* **void** *addSeconds*(**unsigned int** *seconds*);
* **void** *addMilliseconds*(**unsigned int** *milliseconds*);

**Task 1.6**

Implement the following constructor for *DateTime*. If any of the parameters are invalid, use the default values. In no case should the *DateTime* object go invalid.

*DateTime*(**unsigned int** *year* = 1970, **unsigned int** *month* = 1, **unsigned int** *day* = 1, **unsigned int** *hour* = 0, **unsigned int** *minute* = 0, **unsigned int** *second* = 0, **unsigned int** *millisecond* = 0);

**Task 1.7**

Implement the following getter and setter

* **unsigned int** *getYear*().**const**;
* **unsigned int** *getMonth*() **const**;
* **unsigned int** *getDay*() **const**;
* **unsigned int** *getHour*() **const**;
* **unsigned int** *getMinute*() **const**;
* **unsigned int** *getSecond*() **const**;
* **unsigned int** *getMillisecond*() **const**;