

# CSE 459.22 – Programming in C++

## Lab 2

---

**Assigned:** 13<sup>th</sup> Apr, 2006

**Due:** 26<sup>th</sup> Apr, 2006 11:59 PM

**Late Penalty:** 20 points per day

This lab might take long time so do not wait until the last moment. Start as early as possible.

---

### **Goal:**

- To familiarize yourself with coding in Object Oriented Programming
- Getting started with Classes, Objects and their usage

**Points:** This lab is worth **100** points

### **Problem Description**

In this lab, you are going to develop a calendar system which prints out the dates (along with corresponding days) for the given a month and year. User inputs the following details to help you calculate the required month's days:

1. Year – Year for which the calendar is required
2. Month – Month for which you have to output the calendar
3. New Year Day – Day on which Jan 1<sup>st</sup> falls. For example, for 2006, it would be Sunday

Year should be used to check whether it is a Leap year or not. YYYY is leap year if following two conditions hold:

- should be perfectly divisible by 4
- **If** it is divisible by 100 then it **should not** be divisible by 400

New Year Day (NYD) is required so that you can calculate the starting day of the required month i.e., unless you know that year 2006 starts with Sunday, you will not be able to know that May 2006 starts with Monday.

### **Format of Input**

- Year – YYYY (Ex: 1999)
- Month – MM (Ex: 11, 04 or 4. Leading zero is optional i.e., user has a choice to enter either 04 or 4)
- New Year Day (NYD) – One of the following characters
  - o M – Monday
  - o T – Tuesday

- o W – Wednesday
- o R – Thursday
- o F – Friday
- o S – Saturday
- o U – Sunday

Appropriate error should be raised if any of the input is not in the correct format. Examples of bad input are: 99 for year, 15 for month, ‘T’ for NYD.

### Example Input & Output

Year: 2006

Month: 06

NYD: R

Output:

| Su | Mo | Tu | We | Th | Fr | Sa |
|----|----|----|----|----|----|----|
|    |    |    |    | 1  | 2  | 3  |
| 4  | 5  | 6  | 7  | 8  | 9  | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 25 | 26 | 27 | 28 | 29 | 30 |    |

Year: 2008

Month: 4

NYD: T

Output:

| Su | Mo | Tu | We | Th | Fr | Sa |
|----|----|----|----|----|----|----|
|    |    | 1  | 2  | 3  | 4  | 5  |
| 6  | 7  | 8  | 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 |    |    |    |

Note that, NYD can be any day. It need not be the exact starting day of the year. For example, in the first example it is valid to input NYD as Monday though 2006 starts with a Sunday. It should not matter because you will be calculating according to the day given and you do not have any other way to get the starting day of the year (C++ does not provide any functions, as far as I know). While testing your program, you can test with the actual day of the year. If you give arbitrary NYD, then verification of your output can be daunting.

### Suggested Approach

Here is how I would do this problem. It is just to give you a hint. You can do whatever

way you can. Intelligent solutions will enjoy extra credits!! Create an array of 12 numbers which stores the number of days in each month. Here, take care of the leap year! Based on the NYD and this array, calculate the first day of the required month. Once you get that, it is straight forward to printout the calendar. You just have to start with that date and print dates by making sure to insert a new line after every Saturday. You also need to print enough blank spaces so that Date 1 starts from appropriate position in the line. You should also print the header line (Su Mo Tu We Th Fr Sa).

## Requirements of the Program

Though this program can be done without using C++ features or classes, since it is a C++ course, you **SHOULD** use classes and objects to code. To be specific, I need you to have two classes with data members:

- Date
  - nDay – Number (1..31)
  - sDay – Day (U/M/T/W/R/F/S)
- Month
  - Month Number (1..12)
  - noDays # of days (28/29/30/31)
  - Starting day – Day of date 1 of this month
  - Array of objects of Date class (# of objects in array should be noDays)

Note that, you can add as many variables as you want to this list. All I need is a Date class to store the Day's number and day (M/T..) and a class for Month which contains an array of Date objects. You are free to add more information if you think is required.

Each Class should have enough constructors. You can create as many constructors as you want based on your program's design and based on the way you want to initialize the objects. You **SHOULD NOT** depend on the constructor provided by the compiler. At the minimum, you should have at least a default constructor.

You should allocate memory for Array using Dynamic Memory Allocation (**new** operator).

You should de allocate the memory allocated on Heap (using **delete** operator) whenever you are done playing with Month object. (Hint: To make this process simple, you can make use of Destructors).

All the code should present in some member functions of classes. Remember that creation of **Helper Functions** (private functions in each class) and **Public Interface** (public functions in each class) will help you in writing the code that is well organized.

**DO NOT just write C code in just one function of a class. Such programming will result in a large penalty.**

Remember that, I will give enough partial credit based on your work. So, give it a

serious try and put enough effort. And, I am sure you will get a good grade.

You can create as many files as you want. Make sure to create a README file which briefly describes the functioning and design of your program. That will help me understanding your code better. And, write comments in the code wherever you think is required.