Lab 3 - Write an Employee Class

Write a class named `Employee` that has the following member variables:

- **`name`** — a string that holds the employee's name
- **`idNumber`**—an `int` variable that holds the employee's ID number
- **`department`**—a string that holds the name of the department where the employee works
- **`position`**—a string that holds the employee's job title

The class should have the following constructors:

- Write appropriate mutator functions that store values in these member variables and accessor functions that return the values in these member variables. Once you have written the class, write a separate program that creates three `Employee` objects to hold the following data: (Use the driver test program)

Name	ID Number	Department	Position
Susan Meyers	47899	Accounting	Vice President
Mark Jones	39119	IT	Programmer
Joy Rogers	81774	Manufacturing	Engineer

- You must use Makefile or similar to compile multiple cpp files into an executable program.
- You need to define the class object definition in a .h header file. Save the file in employee.h
  - Your employee.h should use

```
#ifndef EMLOYEE_H
#define EMPLOYEE_H
/* define the class object Employee here
*/

#endif
```

- You need to place the class object implementation file in employee.cpp
  - You employee.cpp should include employee.h

```
#include employee.h"
/*
  Define your class object implementation program here
*/
```

- Save the following test driver program to **employee_main.cpp** which contains:

```
// Use the below Driver program to demonstrate the class

#include "employee.h"
#include "printMeFirst.h"
// please include all other header files and anything you need
int main()
{
        printMeFirst ("Ron Sha", "CS – 116: Lab 3 - Employee"); //put your name
        // Create an Employee object to test constructor #1.
        Employee susan;

        susan.setInfo("Susan Meyers", 47899, "Accounting", "Vice President");
        // Create an Employee object to test constructor #2.
        Employee mark ;

        Mark.setInfo("Mark Jones", 39119,"Dept","Title");
        mark.setDepartment("IT");
        mark.setPosition("Programmer");

        // Create an Employee object to test constructor #3.
        Employee joy;
        joy.setName("Joy Rogers");
        joy.setIdNumber(81774);
        joy.setDepartment("Manufacturing");
        joy.setPosition("Engineer");

        // Display each employee's data.
        displayEmployee(susan);
        displayEmployee(mark);
        displayEmployee(joy);

        return 0;
}
```

Your lab should have the following files: (Please refer to Makefile tutoring video posted in canvas)

1. employee_main.cpp (this is the main test driver program from the above)
2. employee.h (this is the definition header file where you define the class object of employee)
3. employee.cpp (this is the implementation file of the definition header file defined in employee.h)

4. printMeFirst.h
5. printMeFirst.cpp
6. Makefile

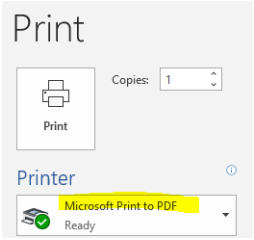
Use the test driver program as test cases for your program **and your program should output similar to the one below:**

```
Course Info: CS - 116: Lab 2
Date: Sun Feb 11 15:03:36 2018

Name: Susan Meyers
ID Number: 47899
Department: Accounting
Position: Vice President

Name: Mark Jones
ID Number: 39119
Department: IT
Position: Programmer

Name: Joy Rogers
ID Number: 81774
Department: Manufacturing
Position: Engineer
```

Lab Submission: See the lab submission requirements published in canvas.

To submit your assignment in canvas, you must submit TWO files (one pdf and one zip) as follows:

1. **Attach pdf file** which contains source codes you have written and program output screenshot so I can easily read in one file. You can use a word editor to place all the required programs and screenshots, and then use 'Print' to 'Microsoft Print to PDF' to save to a pdf file



The pdf file MUST have the following sections (1. Program Description, 2. Program Source Code and 3. Program Output).

1. Program Description

- brief *description* of the purpose of the *program and*
- an explanation of what your software does and what problem it solves

2. Program Source Code

- Include all the source codes (program files) you have written for this lab (screenshots are ok)
- Your program must have adequate documentation for your source codes:
  - Program description – see above on Program Description
  - **You must put the following function headers for each function (the function header MUST be placed just above the function declaration in your source code). For Functions, it must have:**
    - *Function name: name of this function*
    - *Function description: the purpose of this function and how to use it*
    - *@param param_name and what the parameter/argument is used for*
    - *@return what is returned from the function*
- Make sure your screenshots are readable (not too small)

Below is an example of source code
File: printMeFirst.cpp (list the program file individually)

```cpp
//print_me_first_main.cpp
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

/**
 * @Purpose - this function print out the person who wrote the program,
 * and date/time the program run.
 * @parm - name - the author of the program
 * @parm - courseInfo - the name of the course
 * @return - none
 * @author - Ron Sha
 **/

void PrintMeFirst(string name, string courseInfo)
{
    cout <<" Program written by: "<< name << endl; // put your name here
    cout <<" Course info: "<< courseInfo << endl;
    time_t now = time(0); // current date/time based on current system
    char* dt = ctime(&now); // convert now to string for
    cout << " Date: " << dt << endl;
}
```

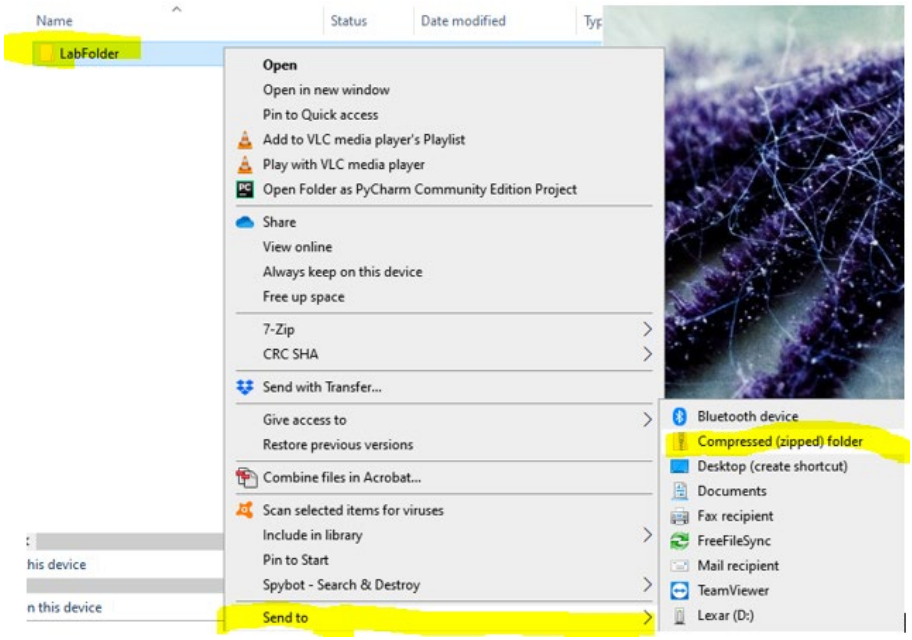
File: list other cpp files separately

3. Program Output

- Attach all the program outputs (screenshots)
- Don't place Source code and program output side by side as it is not readable in screenshot
- Make sure your screenshots are readable (not too small)
- Your main program outputs MUST include your name printout (use the print_me_first function/program).

2. **Attach zip file** which contains all your source code (you can zip the folder) and functions. Even if you only have one source file, you MUST still do a zip file of the folder. I must be able to compile and run your program from all the source code programs after I unzip your zip file.

   You should create a folder for each lab, and place all your programs, functions and all other files related to this lab in this lab folder. To submit the lab folder, you can use "Send to -> compress" in window file explorer to create a zip file of the folder.



3. **Now you can upload both the pdf file and zip file separately as your assignment submission in canvas.**