Gebze Technical University
Department of Computer Engineering
CSE 241/501
Object Oriented Programming
Fall 2015
Homework # 8
Due date Dec 20th 2015

In this homework, you will write classes for computer files in an hierarchy.

`File` is a class that has the following data members

- **`Name`** of the file
- **`owner`** of the file
- **`size`** of the file
- **`Last modificaiton`** time of the file

The class **`File`** has also operations such as

- Setters and getters for the approriate data members
- Function **`path`** that returns the path of the file as string
- Function **`ls`** that prints the file on the screen similar to what "ls file" does. Function **`ls`** also takes parameters similar to Linux ls paramerts -l, -R, -a.
- Function **`cd`** that returns a **`File`** reference if the operation is successful.
- Function **`cp`**, copies the given file to the current file and returns true if sucessful. Otherwise, it returns false.

The are other classes that derive from the **`File`** class. The class **`Directory`** is a special **`File`** that can hold a number of Files in it. It redefines/overrides all necessary functions and adds new data members as needed.

The class **`Executable`** is a File that can be executed. It has members for storing the compiler that created the executable.

The class **`TextFile`** is a File that contains text. It has members for stroring the text type, Ascii, Unicode etc.

An example run can be such as follows:

```
Direcrory mydir("mydir", "owner", ….);
Executable myExec("myExec", ….);
mydir.cp(myExec);
mydir.ls("Rl");
```

would print

```
myExec -rwx akgul 917 7.12.2015
```

Test each of your classes and your functions with several examples and attach your test results. Use real directory structures from Linux environment.

Below are other rules

- You may use STL classes to keep your data
- You may add other data members and functions if needed.
- The class File is the base class. It should not use any information about the derived classes.
- Each class should have its header and implementation files.
- Use name spaces and separation of interface and implementation, and other good programming practices