CIS 246 – C++ Programming

Program:	**Week 11**
Points:	**20**
Chapter(s):	**11**
File(s) to Submit:	**Company.h, Company.cpp, Technology.h, Technology.cpp, Manufacturer.h, Manufacturer.cpp and Portfolio.cpp (within one zip)**

Summary

Write a C++ application that demonstrates inheritance using information about companies.

Description

You will build a portfolio containing the stock prices of 20 companies. The portfolio will be built by flipping a coin and adding a Technology company or a Manufacturing company to the portfolio based on the outcome of the coin flip.

Your program will use three classes to demonstrate inheritance: Company, Technology, Manufacturer and an implementation file as the driver program: Portfolio.cpp. The details of these classes are outlined below.

Learning Objectives

- Inheritance
- Function overriding
- Invoking base class constructors

You must write at least the following three classes and driver program!!

Requirements for the Company Class

Data Members

stock price (as a double)

Be sure to use the appropriate access modifier.

Member Functions

You must use the following constructors and functions:

1. **One-argument constructor** – This class needs only one constructor. This constructor takes a double as an argument and uses it to set the company's stock price. Use best practices in software engineering!

2. **Getter & setter functions** – Provide a getter and setter for each data member as needed. Determine if the getter or setter needs to exist, and, if so, whether it should be public, protected, or private. Use the most restrictive access modifier that is appropriate.
3. **toString function** – This function **DOES NOT PRINT ANYTHING**. It simply builds a string which it then **RETURNS**. Use the ostringstream class, as in the BasePlusCommissionEmployee.cpp example in Fig. 11.15.

Requirements for the Technology Class

The Technology class is derived from the Company class.

Data Members – none

Member Functions

1. **One-argument constructor** – This class needs only one constructor. This constructor takes a double as an argument. It **invokes the constructor of its base class**, passing along the argument.
2. **toString() function** – This class redefines the toString function of its base class, as in the BasePlusCommissionEmployee.cpp example in Fig. 11.15.

   Again, this toString function **DOES NOT PRINT ANYTHING**. It simply builds a string which it then **RETURNS**. The string that it builds includes a portion specific to this class, and a portion that it gets from invoking the toString function of its base class.

Requirements for the Manufacturer Class

The Manufacturer class is derived from the Company class.

Data Members – none

Member Functions

1. **One-argument constructor** – This class needs only one constructor. This constructor takes a double as an argument. It **invokes the constructor of its base class**, passing along the argument.
2. **toString() function** – This class redefines the toString function of its base class, as in the BasePlusCommissionEmployee.cpp example in Fig. 11.15.

   Again, this toString function **DOES NOT PRINT ANYTHING**. It simply builds a string which it then **RETURNS**. The string that it builds includes a portion specific to this class, and a portion that it gets from invoking the toString function of its base class.

Requirements for the Portfolio Program

A **main** function is required. You may include other functions if you prefer. Main builds the portfolio of companies and generates a summary report to the screen.

Main Method Details
Use a counter-controlled loop in main to create 20 companies. Each company will be an instance of either the Technology or the Manufacturing class.

You do not need to use an array to store each object. Just create and print.

For each object to be created:

1. Generate a random integer (or Boolean) to simulate flipping a coin.
2. Create the appropriate object (Technology or Manufacturing) depending on the result of the coin flip. Use this formula to generate the stock price to pass to the constructor of the object: current loop index * 0.11 + 1.
3. Print each Company's details by calling its **toString function.**

Sample Run

Remember you are generating these Companies randomly, so it is unlikely that your output will duplicate this output exactly. Your output should all line up in neat columns.

```
Manufacturer   $1.00
Technology     $1.11
Manufacturer   $1.22
Manufacturer   $1.33
Technology     $1.44
Technology     $1.55
Technology     $1.66
Manufacturer   $1.77
Manufacturer   $1.88
Manufacturer   $1.99
Manufacturer   $2.10
Manufacturer   $2.21
Technology     $2.32
Manufacturer   $2.43
Technology     $2.54
Technology     $2.65
Manufacturer   $2.76
Manufacturer   $2.87
Manufacturer   $2.98
Manufacturer   $3.09
```

Requirements for Full Credit on This Project

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a "better" way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of each source code or header file that includes your name, the assignment number, and a description of the code in that file. There must be comments at each important step in your algorithm that describes that step.

BEST PRACTICES – Follow best practices in C++ programming, including, but not limited to, appropriate use of private/public, appropriate use of classes and/or header files, sets & gets, white space, alignment, meaningful variable names, naming conventions, using statements, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ALL FILES BEFORE THE DUE DATE – See the due date for this assignment on the course calendar in Canvas and review the submission requirements below.

ALL SUBMISSIONS MUST BE YOUR OWN WORK – Review the syllabus regarding plagiarism and the Joliet Junior College Academic Honor Code. Review the Get Help page in every module if you need help with this assignment.

Submission Requirements

All source code file submissions must be compressed into ONE Compressed Folder, or zip file. The dropbox for this assignment will only allow compressed files (.zip extension) to be uploaded.

Instructions for creating a Compressed Folder (zip file) are available here:
https://support.microsoft.com/en-us/help/14200/windows-compress-uncompress-zip-files

Do not submit your code in any other file format but .cpp and .h (within a zip). I will not accept links to online storage. You must submit the actual file. Do not submit executable files. Do not submit project files from an IDE.