COURSEWORK ASSIGNMENT **UNIVERSITY OF EAST ANGLIA**
School of Computing Sciences

Unit : CMP-5015Y
Assignment Title : Assignment 2 — C and C++ Programming (PROVISIONAL)

Date Set	**:** Week 4	
Date & Time of Submission	**:** Thursday week 10 (3 p.m.)	
Return Date	**:**	
Assignment Value	**:** 25%	
Set By	**:** Dr G. C. Cawley	**Signed:**
Checked By	**:** Dr A. Bagnall	**Signed:**

Aim:

The aim of this assignment is for the student to gain experience in the design and implementation of a relatively small program in both C and C++ programming languages. Implementing a similar program in both C and C++ will provide insight in to the development of programming languages.

Learning outcomes:

On successful completion of this exercise, the student will have reinforced a basic understanding of the concepts of classes and objects and will be familiar with the basic syntax of C++ programming constructs used to implement classes (including operator overloading) and have experience with stream-based I/O. The student will also gain familiarity with the more limited forms of data abstraction supported by a more procedural style of programming and the basic elements of the C programming language.

Assessment criteria:

Marking Scheme (out of 100 marks):

- Part 1 - Electronic component inventory system in C [40 marks].
- Part 2 - Electronic component inventory system in C++ [60 marks].

Marks will be awarded according to the proportion of specifications successfully implemented, programming style (indentation, good choice of identifiers, commenting etc.), and appropriate use of programming constructs. Marks may also be awarded for correct use of more advanced programming constructs not covered in the lectures. Program for maintainability - ensure that your programs are easily understood by a human reader, as well as correctly followed by the computer.

Description of assignment:

Part 1 - Simple Electronic Component Inventory System in C

Chartlins is a company that specialises in providing electronic components for hobbyists to build simple electronic components of their own design, or from a kit of parts. They have hired you to write a

system to manage their inventory of components and process sales etc. The program must be written in the C progamming language (conforming to the C11 standard) and demonstrate the use of `structs` and data abstraction to write modular, maintainable programs. The file `inventory.txt` provides details on the companies initial inventory of components (resistors, capacitors, diodes, transistors and integrated circuits). For each stock item, the file provides the component type, the stock code, the number of items in stock and the unit price in pence. The remainder of the record provides some additional information:

- Resistors - the restance in Ohms, encoded according to the BS1852 standard (see `http://www.electronics-tutorials.ws/resistor/res_2.html` for further details).
- Capacitors - the capacitance in Farads, encoded in a manner similar to resistance.
- Transistors - an indication whether they are NPN, PNP or FET devices.
- Integrated Circuits (ICs) - a brief description.

The file `sales.txt` contains a list of sales where customers have bought components from the company. The file describes the date of the transaction, the stock code of the component purchased and the number of items purchased. Your program should apply the transactions described, in this file, generating appropriate error messages if a transaction cannot be completed. After the transactions have been performed, your program should determine the answers to the following queries:

- Print a list of the inventory, sorted in order of increasing price.
- Which day resulted in the greatest sales volume (give the day, the total number of items sold an the total amount of money spent (in pounds and pence).
- How many NPN transistors does Chartlins have in stock after processing all successful sales.
- What is the total resistance of all the remaining resistors in stock?

The program should consist of five files, which define the basis of the marking scheme:

- `StockItem.h` and `StockItem.c` These files should define a `struct` that represents a stock item and a set of functions used to access the `struct` (forming an abstract data type) (10 marks).
- `Inventory.h` and `Inventory.c` These files should define a `struct` representing the inventory of the company as a linked list (or other suitable data structure) and a set of functions used to access the `struct` (8 marks).
- `Sales.h` and `Sales.c` These files should define a `struct` representing the sales data, as a linked list (or other suitable data structure), where each node in the link contains a reference to the StockItem structure, the date and the number of items sold. The list should only record the `successful` transactions (8 marks).
- `StockProgram.c` This file contains the `main` part of the program that loads and processes the inventory and sales data, and provides answers to the four queries (6 marks).
- The remaining 8 marks are awarded for the answers to the four inventory questions listed above.

All questions regarding the specifications must be asked via the appropriate discussion board on the module volume on BlackBoard. Note that `inventory.txt` and `sales.txt` may change before the due date for the assignment, and the answers to the queries must be based on the final versions of these files.

Part 2 - Advanced Electronic Component Inventory System in C++

The second part of the assignment consists of rewriting the electronic component inventory system in C++, demonstrating the full use of the object oriented programming extensions that it provides. The program should use classes, including the use of inheritance, dynamic binding, operator overloading (including stream based I/O) and type conversions.

Again, the program should consist of five files, which define the basis of the marking scheme:

- `StockItem.h` and `StockItem.cpp` These files should define an abstract base class that represents a generic stock item and concrete subclasses representing each type of component. (20 marks).
- `Inventory.h` and `Inventory.cpp` These files should define a `class` representing the inventory of the company as a linked list (or other suitable data structure). The class should have methods that assist in answering the same queries as before (the methods should be generic so they could be used to answer similar queries, rather than specifically those given in the specification) (12 marks).
- `Sales.h` and `Sales.cpp` These files should define classes used to store the sales data, as a linked list (or other suitable data structure). The list should only record the `successful` transactions (12 marks).
- `StockProgram.cpp` This file contains the `main` part of the program that loads and processes the inventory and sales data, and provides answers to the four queries (8 marks).
- The remaining 8 marks are awarded for the answers to the four inventory questions listed above.

Required:

A printed copy of your solution should be submitted to the hub, containing the printouts of your source code followed by a sample of the output from the programs, giving answers to the queries listed above. The files must be submitted in the following order, starting with the C version : `StockItem.h`, `StockItem.c`, `Inventory.h`, `Inventory.c`, `Sales.h` `Sales.c`, `StockProgram.c`, output of C program giving answers to the queries, followed by the C++ version: `StockItem.h`, `StockItem.cpp`, `Inventory.h`, `Inventory.c`, `Sales.h`, `Sales.c`, `StockProgram.c`, and the output of the C++ program. The code must be formatted so that it prints out legibly using a sensible font size (e.g. format code to be no m ore than 80 columns wide) and stapled securely in the top left-hand corner. Submissions where these instructions are not followed will be penalized.

An electronic submission must also be made via BlackBoard, submit a .zip file containing the NetBeans or VisualStudio projects for both parts of the assignment. These must be compilable using the versions of these IDEs available on standard lab computers. If you develop your solution on another platform, you are required to ensure that it compiles and operates correctly under this versions of the IDEs available in the labs. Submissions not conforming to these requirements will be penalized.

Handing in procedure:

A printed copy of your submission must be submitted to the Teaching Hub, by the date indicated above. A .zip file containing the projects for your programs, compiled using the IDEs installed on lab PCs, must also be submitted via BlackBoard.

Plagiarism:

Plagiarism is the copying or close paraphrasing of published or unpublished work, including the work of another student without the use of quotation marks and due acknowledgement. Plagiarism is regarded as a serious offence by the University and all cases will be reported to the Board of Examiners. Work that contains even small fragment of plagiarised material will be penalised.