

Assessment Practical/Observation

Student Name		CIT Number	
Competency Title, Code and Banner Code CRN	C++ Programming ICTPRG414 Apply introductory programming skills in another language ICTPRG415 Apply skills in object-oriented design 34404, 34405 & 34408, 34410		
Assessment Type	<input checked="" type="checkbox"/> Written <input checked="" type="checkbox"/> Project/Programming <input type="checkbox"/> Presentation <input type="checkbox"/> Other		
Assessment Name	Simulated Workplace Task 2: Project		
Assessment Date	Friday the 15 th November, 11pm.		
Student Statement: This assessment is my own work. Any ideas and comments made by other people have been acknowledged. I understand that by emailing or submitting this assessment electronically, I agree to this statement.			
Student Signature		Date	
PRIVACY DISCLAIMER: CIT is collecting your personal information for assessment purposes. The information will only be used in accordance with the CIT Privacy Policy.			
Assessor Feedback (also complete observation checklist and questions on last page) As this assessment is submitted in eLearn, there is no requirement for you to put your signature on this coversheet. <input type="checkbox"/> Student provided with feedback			
Attempt 1	<input type="checkbox"/> Satisfactory	<input type="checkbox"/> Not Yet Satisfactory	Date / /
Attempt 2	<input type="checkbox"/> Satisfactory	<input type="checkbox"/> Not Yet Satisfactory	Date / /
Assessor Name	Christy Rawsthorne	Assessor Signature	
Assessor Note: Please record any reasonable adjustment that has occurred for this assessment.			

Assessment Task Instructions for Students:

You will be required to create a C++ program that show good programming techniques, implements sorting and searching algorithms, has functions and reads data in from a file.

Covering the following topics:

- Object oriented programming techniques showing inheritance
- Standard and File Input/Output
- Program control statements
- Arrays
- Functions
- UML & SDLC

Time Allowed: Friday the 15th November; 11pm

Assessment Range and Conditions: Assessment can be completed in class, or at home. While I expect you to troubleshoot issues with your code, please ask for assistance when if you are having troubles resolving them. If you work with another student, you need to include in the top comments area of your program, their full name.

Materials Provided: this assessment paper, and text file for data input. Visual Studio 2019 is provided in the classroom machines, and the community edition is available to download for PC users.

Information for Students: You may have two (2) attempts for this assessment.

- If your **first** attempt is not successful, your teacher will discuss your results with you and will arrange a second attempt.
- If your **second** attempt is not successful, you will be required to re-enrol in this unit. Only one re-assessment attempt will be granted for each assessment item.

Assessment Criteria: To achieve a Satisfactory result, your assessor will be looking for your ability to demonstrate the key skills/tasks/knowledge detailed in the Assessment Task to industry standard.

- Program compiles without errors
- No global variables used (unless constant value set)
- Base class created. Syntax correct
- Derived class created. Syntax correct
- Class definitions written using correct layout and syntax (and shows good programming techniques).
- Expected output to the screen. Calculations correct
- Expected output in output.txt. Calculations correct
- Insertion and Deletion works as expected.
- Output for file and screen neat and displayed similar to sample output (or what you think is just as nice)
- Input & Output Files open and close correctly. Error message displays and the program exits elegantly if either file is not found.
- Constructors & Destructors used correctly
- Static class diagram created showing generalisation and specialisation.
- Class diagram applies the principles of aggregation and composition.
- Activity Diagram created.
- Sequence Diagram created

All questions satisfactorily answered.

Scenario

Biblioden is a small locally owned bookstore. They would like for you to write a program to help with the management of their booklists. They have been primarily a bricks and mortar style sort, but know they need to start moving towards a computerised system.

Information on the books that they have on hand has been converted into a text file called **products.txt**. You can make the assumption that this text file will never have more than **25** lines of text, but it is always in the same format. [An array of objects will need to be created to store the contents of this file]. The store clerks need to be able to add another book. You will need to ask the clerk to enter all of the information that is usually provided from the text file, and this new book will need to be inserted at the front [index 0] of the array.

The clerk should also be given the option to delete any object from the array [the clerk will only remove a maximum of 1 book from the array].

Finally, you will also be provided with another text file: **orders.txt**. This text file contains a list of books that have been requested via phone from customers of Biblioden. You will need to check the books listed in **orders.txt** against your array of objects and output to the screen whether the books are currently available with enough stock [please see sample output 3].

You are given 2 text files; **products.txt** and **orders.txt** which you can download from eLearn.

products.txt contains information about books on sale at *Biblioden*. The structure is

author surname, book title, stock level, book format, ISBN number, wholesale price

orders.txt contains information about orders received by the bookshop. The structure is:

book title, number on order

The book format is as follows:

a	audiobook
e	ebook
h	hardback
s	paperback / softback

Requirements

You are required to create a Visual C++ program to read the products file and store the information in an array.

- ☐ Create a *base class* called **Product**. This contains the variables **title**, **surname**, **isbn** and **wholesalePrice**. Set these as private member variables with appropriate accessor functions to set and get the data (you can set more than one value with a function).
- ☐ Create a *derived class* called **Stock** based on **Product**. Its additional attributes are **retailPrice**, **book format** and **stockLevel**. These are public.

- ☐ To determine the retail price of each book use the following
 - Audiobook = wholesale price + 43%
 - ebook = wholesale price + 8%
 - hardback = wholesale price + 45%
 - paperback / softback = wholesale price + 27%

- ☐ In the main() function, create an array of objects of the derived class and input all the information into this array from the text file **products.txt**.

- ☐ Ask the clerk to enter in 1 book, and insert this book at the beginning of the array.

- ☐ Ask the clerk if they would like to remove a book from the list. They can remove up to 1 book from the array.

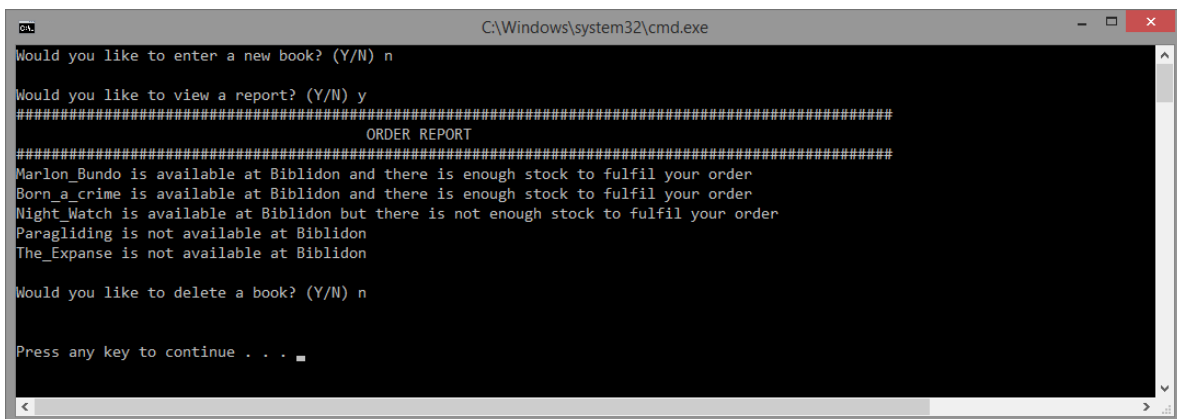
- ☐ Pass this array to **another function** that writes the member variables in a neat format to an output file **report.txt** (see example below).

- ☐ Read the **orders.txt** file and search the array of stock objects for a match on book title. If a match is found, display the order details to the screen as shown below, noting if the order can be filled (that is, is the book available and is the stock level sufficient).

As well as producing the output specified, your program **must** include the following features:

- Ensure you include comments where appropriate
- Include some code to exit the program gracefully if the text files cannot be found.

Sample program screenshot:



```

C:\Windows\system32\cmd.exe
Would you like to enter a new book? (Y/N) n

Would you like to view a report? (Y/N) y
#####
                        ORDER REPORT
#####
Marlon_Bundo is available at Biblidon and there is enough stock to fulfil your order
Born_a_crime is available at Biblidon and there is enough stock to fulfil your order
Night_Watch is available at Biblidon but there is not enough stock to fulfil your order
Paragliding is not available at Biblidon
The_Expense is not available at Biblidon

Would you like to delete a book? (Y/N) n

Press any key to continue . . .
  
```

Sample Report.txt

report.txt - Notepad

File Edit Format View Help

<< Biblioden Books >>

Audiobooks:

ISBN	title	author	Retail Price	Stock Level
9780575102484	The_way_of_kings	Sanderson	\$24.24	10
9780732298241	Golden_Lion	Smith	\$27.17	3
9780141805917	The_BFG	Dahl	\$16.06	7
9781930446458	Night_Watch	Pratchett	\$44.02	2

eBook:

ISBN	title	author	Retail Price	Stock Level
9780857054036	Girl_with_the_dragon_tattoo	Larsson	\$17.23	5
9780006486121	A_Feast_for_Crows	Martin	\$10.79	20
9781841499895	Leviathan_Wakes	Corey	\$12.95	2

Hardcover:

ISBN	title	author	Retail Price	Stock Level
9781591025948	The_Blade_Itself	Abercrombie	\$62.55	12
9780091956141	The_Martian	Weir	\$47.85	6
9780356500904	Skin_Game	Butcher	\$35.45	1
9780575104662	Calamity	Sanderson	\$53.00	10
9781452173801	Marlon_Bundo	Twiss	\$16.52	30
9780439708180	Goblet_of_Fire	Rowling	\$10.14	1

Softcover:

ISBN	title	author	Retail Price	Stock Level
9780007444229	Fools_Quest	Hobb	\$25.34	2
9780575081406	The_Name_Of_the_Wind	Rothfuss	\$23.56	10
9781406328899	Wheres_Wally	Handford	\$6.99	0
9780071808552	Java	Schildt	\$79.95	4
9780385689229	Born_a_crime	Noah	\$44.45	13
9780538798082	C++	Malik	\$53.98	0
9780061146305	The_Bad_Beginning	Snicket	\$6.50	9
9780486415864	Great_Expectations	Dickens	\$10.15	2

Object-Oriented Design diagrams that need to be produced:

1. Throughout the semester, during the lecture there has been a number of *Good Programming Techniques* suggested. Describe what you have done in your programming that shows good programming techniques and how this has helped your program.
2. Name a specific SDLC that would be appropriate to use for this project. Why do you think it would be appropriate?

3. Provide 2 test cases:

Verify that the products.txt file exists	Create an ifstream operator and open the file	The file will open, or it will elegantly exit the program	The file was in the wrong location and the program exited without crashing.

4. Using the provided scenario, create an UML class diagram that shows high cohesion. It should also show generalisation and specialisation. You can adapt the classes that I have asked you to implement for the c++ portion of the assessment [I mean that the diagram should show high cohesion, etc, but this does not have to be the class that you implemented].
5. Create an UML object diagram showing an example of what the clerk could insert.
6. Create an UML activity diagram that shows what happens when the clerk needs to insert or delete a new book from the array of objects.
7. Create an UML sequence diagram that shows how the objects in the array interact with the objects from the *orders.txt* file.