Ch27

Object-Oriented Programming (OOP)

CHAPTER ASSIGNMENT · 45 MARKS

Name _____ Class _____ No. _____ Date _____

Answer every question in the space given. Show your working. You may use the chapter page to help you.

Questions from *Computer Science for Cambridge International AS & A Level Coursebook*, 2nd edition, Sylvia Langfield and Dave Duddell (Cambridge University Press), for classroom use.

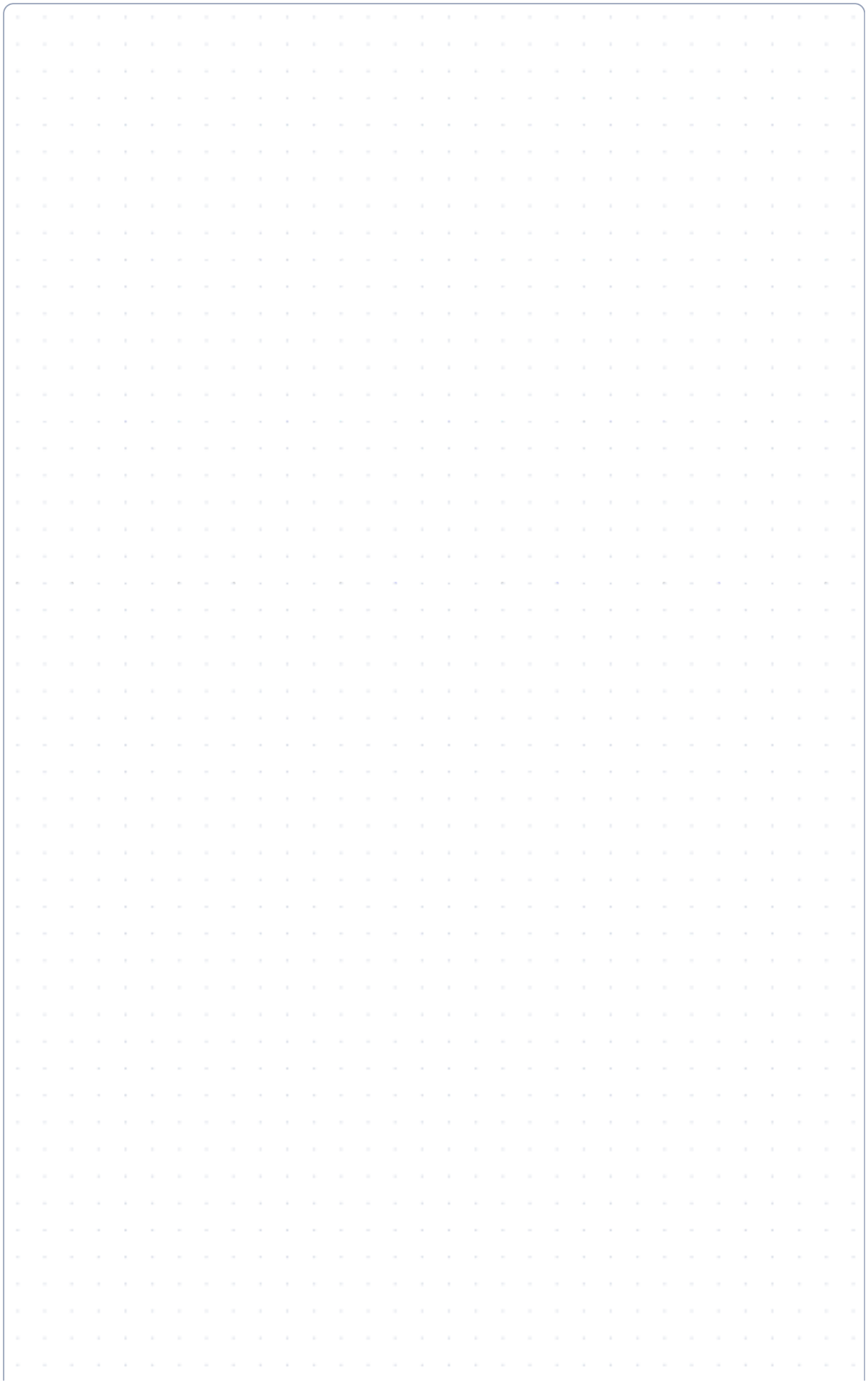
Tasks and questions from the chapter

27.01

(1) Copy the `Car` class definition into your program editor and write a simple program to test that each method works.

(2) A business wants to store data about companies they supply. The data to be stored includes: company name, email address, date of last contact.

(a) Design a class `Company` and draw a class diagram.

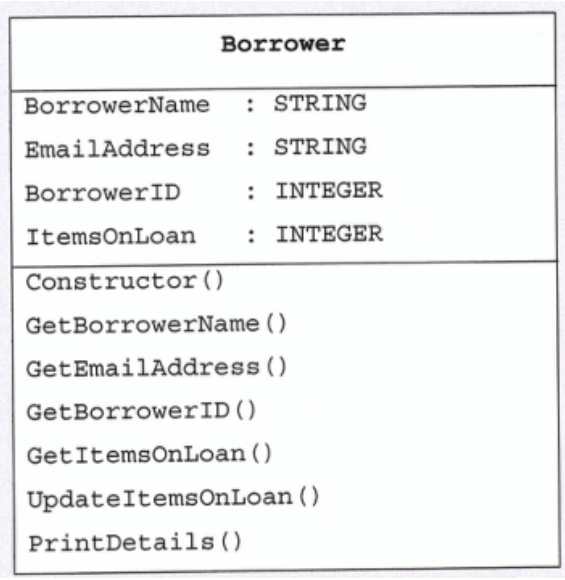


(b) Write program code to declare the class. Company name and email address are to be set by the constructor and will never be changed.

(c) Instantiate one object of this class and test your class code works.

27.02 Copy the class definitions for `LibraryItem`, `Book` and `CD` into your program editor. Write the additional get methods. Write a simple program to test that each method works.

27.03 Write code to define a Borrower class as shown in the class diagram in Figure 27.05.



The constructor should initialise `ItemsOnLoan` to `0`.

`UpdateItemsOnLoan()` should increment `ItemsOnLoan` by an integer passed as parameter.

Write a simple program to test the methods.

27.04 Use your program code from Task 27.02. Re-define the `PrintDetail` methods for the `Book` class and the `CD` class. Test your code.

27.05 Use your program code from Task 27.03. Add another attribute, `BorrowerID`, to the `LibraryItem` class so that the item being loaned can have the borrower recorded.

Change the `LibraryItem.Borrowing` and `LibraryItem.Returning` methods, so that `LoanItem.BorrowerID` and `Borrower.ItemsOnLoan` are updated when a library item is borrowed or returned.

27.06 Use your code from Task 27.02 or Task 27.04. Add another attribute, `RequestedBy`, to the `Book` class so that the borrower making the request can be recorded.

Change the method `Book.SetIsRequested`, so that `Book.RequestedBy` is updated when a book is requested.

27.07 Use your code from Task 27.06 to write the complete program to implement a simplified library system.

Write code to provide the user with a menu to choose an option. An example of a menu that would be suitable is shown in Figure 27.06.

```
1 - Add a new borrower
2 - Add a new book
2 - Add a new CD
4 - Borrow a book
5 - Return a book
6 - Borrow a CD
7 - Return a CD
8 - Request book
9 - Print all details
99 - Exit program
Enter your menu choice:
```

27.08 Write the code required for the `Lesson` and `Assessment` classes. Add the code for the `Course` class and test your program with the appropriate simple program from Worked Example 27.03.

RP **Reflection Point:** Explain why a class can be regarded as a user-defined type.

Exam-style questions

1 A program is to be written using an object-oriented programming language. A bank account class is designed. Two subclasses have been identified:

• **Personal Account**: the account holder pays a monthly fee and may overdraw the account up to an agreed overdraft limit.

• **Savings Account**: the account holder must maintain a positive balance and gets paid interest on the balance at an agreed interest rate.

(a) Draw an inheritance diagram for these classes. [3]

(b) The design for the `BankAccount` class consists of:

- attributes: `AccountHolderName`, `IBAN` (International Bank Account Number)
- methods: `CreateNewAccount`, `SetAccountHolderName`, `GetAccountHolderName`, `GetIBAN`

Write **program code** for the class definition of the superclass `BankAccount`. [5]

(c) State the attributes and/or methods required for the subclass `PersonalAccount`. [4]

(i)

(ii) State the attributes and/or methods required for the subclass `SavingsAccount`. [4]

(iii) Identify the feature of object-oriented program design that combines the attributes and methods into a class. [1]

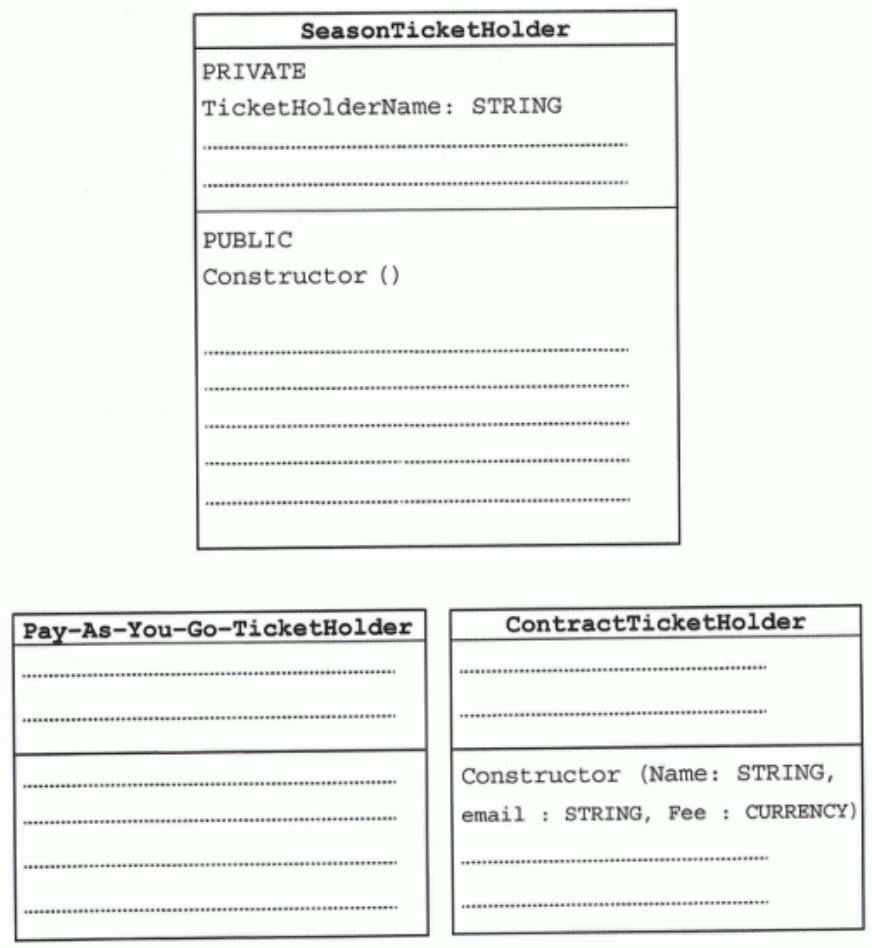
2 A bus company in a town has two types of season ticket for their regular customers: pay-as-you go and contract. All season ticket holders have their name and email address recorded.

A pay-as-you-go ticket holder pays a chosen amount into their account. Each time the ticket holder makes a journey on the bus, the price of the fare is deducted from the amount held in the account. They can top up the amount at any time.

A contract ticket holder pays a fixed fee per month. They can then make unlimited journeys on the bus.

The bus company wants a program to process the season ticket data. The program will be written using an object-oriented programming language.

(a) Complete the class diagram showing the appropriate attributes and methods. [7]



(b) Attributes and methods can be declared as either public or private.

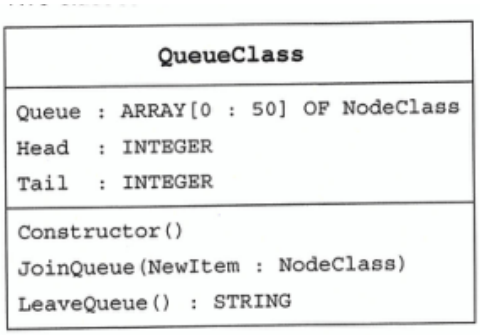
(i) Explain why the `SeasonTicketHolder` attributes are declared as private. [2]

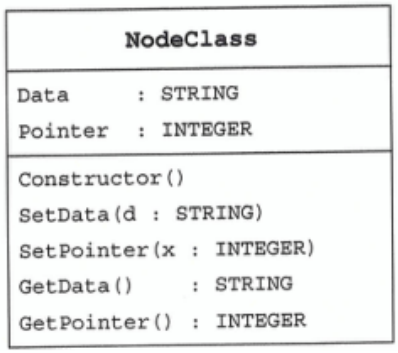
(ii) Explain why the `SeasonTicketHolder` methods have been declared as public. [2]

(c) Write program code to create a new instance of `ContractTicketHolder` with: [3]

- Identifier: `NewCustomer`
- name: A. Smith
- email address: xyz@abc.xx
- monthly fee: $10

3 A queue abstract data type (ADT) is to be implemented using object-oriented programming. Two classes have been identified: `Queue` and `Node`. The class diagrams are as follows:





(a) State the relationship between these two classes. [1]

(b) The `NodeClass` constructor is to: **[10]**

- create a new node
- initialise the `Data` attribute to the empty string
- initialise the `Pointer` attribute to -1.

Write program code to define `NodeClass`, including the get and set methods.

(c) The `QueueClass` constructor is to: **[3]**

- create a new queue
- initialise the `Head` and `Tail` attributes to -1.

Write program code to define the constructor for `QueueClass`.

(d) The `JoinQueue` method is to: [5]

- create a new object, `Node`, of `NodeClass`
- assign the value passed as parameter to the `Data` attribute of `Node`
- assign `Node` to the end of `Queue`.

Write program code to define the `JoinQueue` method.

End of assignment · Check your answers against the chapter at gurujazzy.com/subjects/computer-science/as-a-level/object-oriented-programming/