B3.1

OOP for a single class

CHAPTER ASSIGNMENT

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 the IB Diploma*, 1st edition (2025), Paul Baumgarten, Ioana Ganea, Carl Turland (Hachette Learning), for classroom use.

Linking Questions (pp. 395, 401, 405)

1. Is OOP necessary for all programming or just in the modelling of complex situations? (B2)

2. How can the principles of encapsulation and information hiding (B2.5) be applied to secure network communication? (A3)

3. In what ways can OOP (B1) be applied to database (A3) development?

Review Questions (pp. 412–413)

1. Which of the following is NOT an advantage of using OOP?

   (a) Reusability of code

   (b) Easy to debug

(c) Improved performance in all scenarios

(d) Encapsulation of data

2 What does abstraction in OOP help with?

(a) Removing all bugs from an application

(b) Hiding complex implementations behind simpler interfaces

(c) Decreasing the use of memory

(d) Making code public to all classes

3 In UML Class diagrams, there is a three-row box used to represent a Class. What does the second row contain?

(a) Class name

(b) Methods

(c) Properties

(d) Statics

4 Which of the following is true about methods in a Class diagram?

(a) They are depicted with a minus sign (`-`) for private methods

(b) They cannot accept parameters

(c) They must be static

(d) They are optional

5 Which statement is true about static methods in Java?

(a) They can be called on instances of a Class

(b) They are called on the Class itself, not the instance

(c) They can directly access and modify instance variables

(d) They must return a value

6 What is a non-static variable also known as?

(a) Class variable

(b) Local variable

(c) Global variable

(d) Instance variable

7 What is the purpose of a constructor in a Class?

(a) To declare variables

(b) To initialize an object

(c) To clean up resources

(d) To return data

8 Which of the following declarations correctly creates an array of objects in Java?

(a) `ClassName[] arrayName = new ClassName[5];`

(b) `ClassName arrayName[] = new ClassName(5);`

(c) `ClassName arrayName = new ClassName[5];`

(d) `ClassName[] arrayName = new ClassName();`

9 Which of the following best describes encapsulation?

(a) Storing data in public fields

(b) Combining data and methods that operate on the data into a single unit or Class

(c) Dividing code into various functions

(d) Making all methods static

10 What are access modifiers used in OOP for?

(a) To define how variables can be modified

(b) To name methods and variables

(c) To indicate static methods only

(d) To control the visibility of Class members

11 Describe one disadvantage of using OOP.

12 Explain how encapsulation can benefit a software project.

13. Explain why a static method cannot access non-static fields or methods.

14. Explain the role of the `new` keyword in object creation.

15. Describe an example of how `private` access modifiers contribute to information hiding.

Programming Exercises (pp. 414–416)

1. **Library system:** Create a simple library system where each book can be either available or borrowed. Implement methods to manage the state of each book and track the total count of books in the library.

```
class Book {
    private static int bookCount = 0;
    private String title;
    private boolean isBorrowed;

    public Book(String title) {
        this.title = title;
        this.isBorrowed = false;
        bookCount++;
    }
    public void borrow() {
        if (!isBorrowed) {
            isBorrowed = true;
            System.out.println(title + " has been borrowed.");
        } else {
            System.out.println(title + " is already borrowed.");
        }
    }
    public void returnBook() {
        // Implement logic to mark the book as not borrowed
    }
    public static int getBookCount() {
        return bookCount;
    }
    public String toString() {
        return title + " - " + (isBorrowed ? "Borrowed" : "Available");
    }
}
```

(a) Complete the `returnBook` method.

(b) Instantiate more books and simulate borrowing and returning books.

(c) Print the status of all books and the total book count.

2 **Simple inventory system:** Create a basic inventory management system for a store. Each product has an ID, name, price and quantity. Implement functionality to add products, update inventory quantity and list all products.

```
class Product {
    private static int nextProductId = 100;
    private int productId;
    private String name;
    private double price;
    private int quantity;

    public Product(String name, double price, int quantity) {
        this.productId = nextProductId++;
        this.name = name;
        this.price = price;
        this.quantity = quantity;
    }
    public void updateQuantity(int amount) {
        // Implement logic to update product quantity
    }
}
```

(a) Implement the `updateQuantity` method in the `Product` class to adjust the stock of a product.

(b) Test adding products, updating quantities and listing all products to ensure the inventory displays correctly.