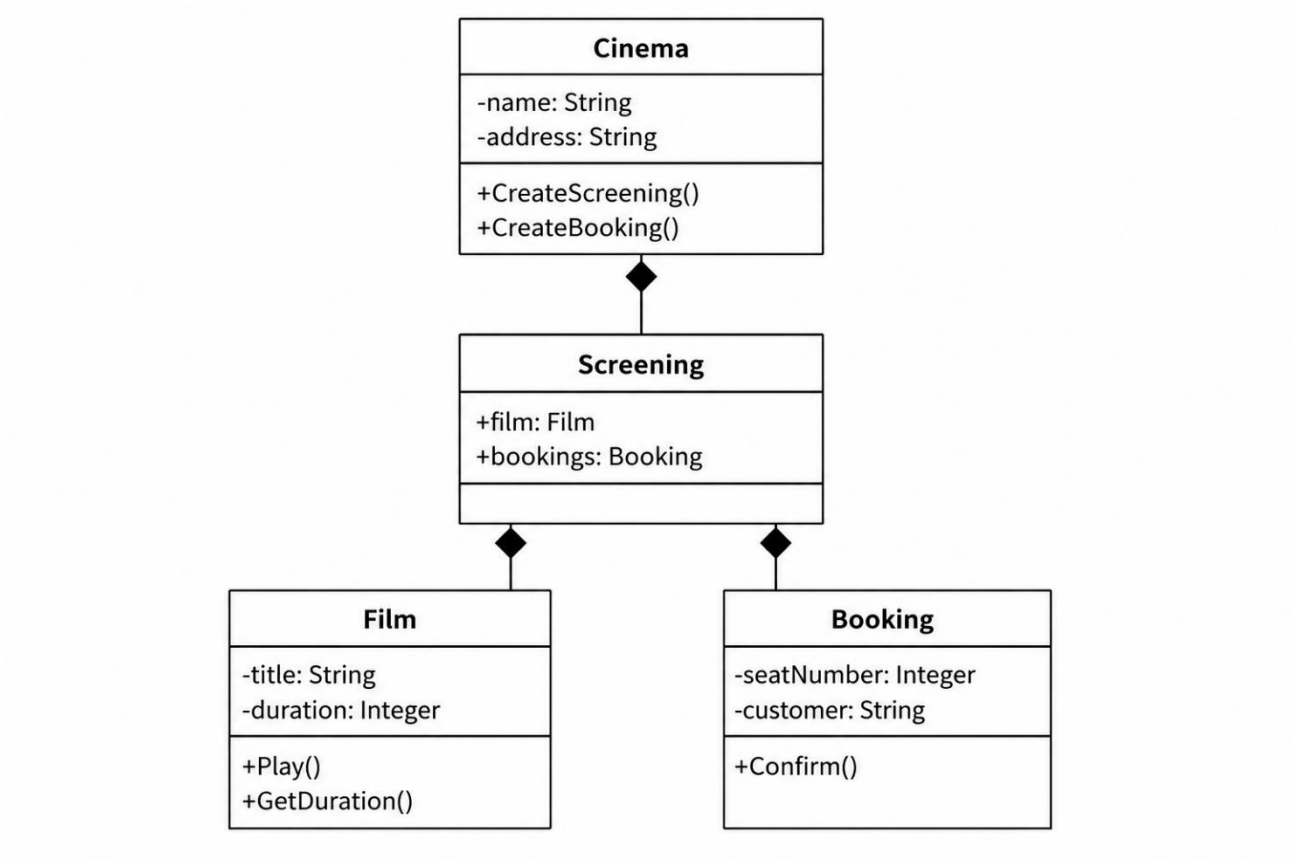
4.1.2 Programming Paradigms

1) The class diagram in **Figure 1** is a representation of relationships between some classes in a program.

Figure 1



a) Write Yes or No in the cells in **Table 1** to identify if the given feature is present in the class diagram shown in **Figure 1**.

Table 1

Feature	Is present in Figure 1? (Yes/No)
Inheritance	**No**
Protected method	**No**
Private attribute	**Yes**

[3 marks]

b) Aggregation and composition are two of the types of relationship that can exist between classes.

Which of these two types of relationships is **not** shown in **Figure 1**?

[1 mark]

- **Aggregation**

c) Explain what the composition relationship between `Cinema` and `Screening` means.

[2 marks]

- **Cinema contains/owns a Screening object**
  // a Cinema has a Screening;
- **If the Cinema is deleted, its Screening objects are also deleted**
  // cannot exist independently;

d) Explain the difference between a protected attribute and a private attribute.

[2 marks]

- **A protected attribute can be accessed within its class and by derived classes/subclasses;**
- **A private attribute can only be accessed within its class;**

e) In the `Film` class there is an attribute `duration` and a method `GetDuration`.

Explain the need for the `GetDuration` method and explain why this approach is favoured in object-oriented programming.

[2 marks]

- **`duration` is a private attribute and is not accessible outside of the `Film` class;**
  // `GetDuration` is a public method and is accessible outside of the `Film` class;
- **This means the way `GetDuration` is represented can be modified without having to change any other objects that interact with `Film`;**
  // easier to reuse/inherit from the `Film` class;

2) A structured programming approach can be used in the production of a program.

a) Explain what is meant by a structured programming approach.

[2 marks]

- **Decomposition (of a problem/program) // use of top-down approach // Structured programming makes use of subroutines / modules;**
- **Use of block structures / compound statements;**
- **Structured programming makes use of control structures; A. by example - sequence / selection / iteration;**
- **Avoidance of use of goto statements;**

b) Explain **three** reasons for adopting the structured approach.

[3 marks]

- **Can get an overview of (the structure of) the program // code is easier to understand;**
- **Can break problem down into sub-tasks;**
- **Can re-use subroutines/modules; A. less duplication of code**
- **Can distribute (implementation of) subroutines/modules among team;**
- **Can test subroutines/modules independently // quicker/easier to debug/maintain // easier to locate errors;**

3) In object-oriented programming:

a) What is meant by polymorphism?

[1 mark]

- **A method shared (up and down the inheritance hierarchy chain) but with each class / method implementing it differently**
  // A single interface is provided to entities/objects of different classes / types
  // Objects of different classes / types respond differently to the use of a common interface / the same usage
  // Allowing different classes to be used with the same interface
  // The ability to process objects differently depending on their class / type;

b) What is meant by overriding?

[1 mark]

- **When a derived class / subclass has a different implementation for a method/function/subroutine to the class it inherits from/from the base class;**

c) When a derived class overrides a method from a base class, the base class method could be either a virtual method or an abstract method.

Describe **two** differences between a virtual method and an abstract method.

[2 marks]

- **Virtual methods can (but do not have to) be overridden by the derived class while abstract methods must be overridden by the derived class;**
- **Virtual methods have an implementation/body while abstract methods do not contain an implementation/body;**
- **Abstract methods can only be declared in abstract classes while virtual methods can be declared in abstract and non-abstract classes;**