KK07 — Principles of OOP

Programming · Comprehensive exam study guide

STUDY DESIGN — YOU MUST KNOW

- abstraction
- encapsulation
- generalisation
- inheritance

Object-oriented programming organises a solution into objects. You must define each of the four principles and recognise them in a scenario — these definitions are high-value, frequently-examined marks.

The four principles

Principle	Meaning	Example
Abstraction	Hiding complex detail and showing only the essential features needed to use something.	A `Car` object exposes `drive()` without revealing engine internals.
Encapsulation	Bundling data (attributes) and the methods that act on it inside an object, and restricting direct access to that data.	A `BankAccount` keeps `balance` private; you change it only via `deposit()`/`withdraw()`.
Generalisation	Identifying shared features and moving them into a common (parent) class.	`Student` and `Teacher` share `name`/`login()`, generalised into a `Person` class.
Inheritance	A subclass (child) acquires the attributes and methods of a parent class and can add/override its own.	`Student` inherits from `Person` and adds `studentID`.

How they relate

Generalisation produces a parent class; **inheritance** is the mechanism by which children reuse it. **Encapsulation** protects an object's data; **abstraction** hides complexity behind a simple interface. Together they make solutions easier to maintain and extend.

REMEMBER

Easy way to keep two apart: **encapsulation** = keeping data and methods together and protected (the 'capsule'); **abstraction** = hiding detail and showing only what's needed.

Key definitions to memorise

Abstraction	Hiding unnecessary detail and exposing only the essential features needed to use something.
Encapsulation	Bundling an object's data and methods together and restricting direct access to the data.
Generalisation	Extracting common attributes/methods into a shared parent class.

Inheritance	A subclass acquiring the attributes and methods of a parent class, with the ability to extend or override them.

Exam technique

EXAM TIP

- Memorise one crisp definition AND one example per principle — examples earn marks in ‘explain’ questions.
- If a scenario shows a parent–child class pair, the principle is inheritance/generalisation; if it shows hidden internals, it's abstraction/encapsulation.

COMMON MISTAKES

- Swapping abstraction and encapsulation definitions — the most common error.
- Saying inheritance 'copies' code — it acquires/reuses parent members.

Quick revision checklist

- ■ I can define all four principles precisely.
- ■ I can give an example of each and identify them in a class scenario.

Exam practice — questions & model answers

Q1. Define encapsulation.

Define · 1 mark

Model answer

- Bundling an object's data and the methods that act on it together, restricting direct access to the data (1).

Q2. Explain the difference between generalisation and inheritance.

Explain · 2 marks

Model answer

- Generalisation extracts common features into a shared parent class (1);
- inheritance is the mechanism by which a subclass acquires those parent features (1).

Q3. A `Vehicle` class has `Car` and `Truck` subclasses that reuse its `startEngine()` method and add their own. Name and explain the two OOP principles shown.

Name/Explain · 4 marks

Model answer

- Generalisation — shared features (startEngine) are placed in the parent Vehicle class (2).
- Inheritance — Car and Truck acquire Vehicle's members and extend them with their own (2).