# CS360 – Concepts of Object-Oriented Programming Languages

## Final Exam – 2025

**Time:** 100 minutes
**Lecturer:** Tuan-Dung CAO

> ⚠ Copying may result in receiving a mark of **0**.

---

## Question 1 (15 pts)

### Multiple-Choice Questions

### a) Method Overloading

Two or more methods in a class may have the same name, as long as they differ in:

A. Their return values
B. Their access specifiers
C. Their parameter lists
D. Their memory addresses

---

### b) Inheritance and Access Levels in C++

Consider the following C++ class hierarchy:

```
class Base {
public:
    int a;
protected:
    int b;
private:
    int c;
};

class DerivedPublic    : public Base {};
class DerivedProtected : protected Base {};
class DerivedPrivate   : private Base {};
```

Which of the following statements is **CORRECT** regarding the access levels of inherited members in the derived classes?

A. In `DerivedProtected`, both members `a` and `b` become `protected`;
in `DerivedPrivate`, both members `a` and `b` become `private`.

B. In all derived classes (`DerivedPublic`, `DerivedProtected`, `DerivedPrivate`),
member `c` is directly accessible from the member functions of the derived class.

C. In `DerivedPublic`, member `a` is `public`, member `b` is `protected`,
and member `c` is `private`.

D. Private inheritance in `DerivedPrivate` converts **all members** (`a`, `b`, and `c`)
of `Base` into `private` members of the derived class.

---

# Question 2 (20 pts)

## Encapsulation in OOP

Present the **advantages (benefits)** of **encapsulation** in Object-Oriented Programming Languages (OOPL).

- Clearly explain each benefit.
- Illustrate each benefit with a **specific source code example** (Java or C++).
- Provide a brief explanation of how your code demonstrates that benefit.

---

# Question 3 (15 pts)

## OOP Principles

Based on your understanding of the **main principles of Object-Oriented Programming**, explain the following statement:

> *"Systems built using object-oriented technologies are **flexible to change** and provide opportunities to create and implement **reusable components**."*

---

# Question 4 (50 pts)

## Preferred Customer Plan

A retail store offers a **Preferred Customer Plan**, where customers earn discounts on all their purchases.
The discount rate is determined by the customer's **cumulative purchase total** as follows:

- When cumulative spending reaches **$500**, the discount is **5%**
- When cumulative spending reaches **$1,000**, the discount increases to **6%**
- When cumulative spending reaches **$1,500**, the discount increases to **7%**
- When cumulative spending reaches **$2,000 or more**, the discount increases to **10%**

---

## a) (20 pts) – Customer Class (Java)

Using **Java**, develop a class named `Customer` with the following fields:

- Customer name
- Telephone number
- Customer number
- A boolean indicating whether the customer wishes to be on a mailing list

Requirements:

- Write one or more constructors

- Implement appropriate **mutator (setter)** and **accessor (getter)** methods

---

## b) (15 pts) – PreferredCustomer Class

Write a class named `PreferredCustomer` that **extends** the `Customer` class.

Additional fields:

- Amount of purchases
- Discount level

Requirements:

- Write one or more constructors
- Implement appropriate mutator and accessor methods
- Override the `toString()` method to return a string in the following format:

```
Customer Name - Phone Number - Customer Number - Amount of Purchases - Discount
Percentage
```

---

## c) (15 pts) – CustomerManage Class

Develop a class named `CustomerManage` to demonstrate the functionality of the classes above.

Requirements:

- Maintain a collection of customers using an **array of objects** (e.g., `Customer[]`)
- Implement a method (e.g., `addCustomer`) to add a `Customer` or `PreferredCustomer` to the array

In the `main` method:

- Instantiate and add the following objects to the collection:

```
Customer:
Jason Walker ("Jason Walker", "0982345211", "01123221", true)

Preferred Customer:
Bob Dawson ("Bob Dawson", "0982345211", "01123221", false, 1800, ...)
```

- Display information for **all stored customers** using their `toString()` methods