Ministry of Higher Education and Scientific Research
University of Tissemsilt
Faculty of Sciences and Technology
Department of Mathematics and Computer Sciences

Exam: Object-Oriented Programming **Domain / Field / Specialty:** Math & Info / Computer Science / ISIL
Academic Year: 2023 / 2024 **Semester / Session:** S05 / SN **Date:** 04/05/2024

Part I: Questions

1. What is the difference between an object and a class in OOP?
2. What does encapsulation achieve in OOP?
3. What is the significance of the ***final*** keyword in Java?
4. What is the primary purpose of a constructor in a class?
5. Can a Java class have multiple constructors?
6. Describe a scenario where the ***this*** keyword is necessary in a constructor or method. Provide a code example

Part II: Practical

Design and implement an object-oriented program to automate the management of a library system that includes books and members.

1) Person Class :
   - Attributes: Name, firstName , age
   - Method: display(): Prints the name, first name, and age.

2) Member Class (inherits from Person)
   - Attribute: memberNumber
   - Method: display()

3) Author Class (inherits from Person) :
   - Attribute: authorNumber
   - Method: display()

4) Book Class :
   - Attributes: ISBN, title, authors
   - Method: display()

5) Library Class :
   Main Method:
   - Instantiate a Member.
   - Instantiate a Book with two Authors.
   - Display the details of the Member, Authors and the Book.

Exam correction

Exam correction : Object-Oriented Programming	**Semester / Session:** S05 / SN	**Date:** 04/05/2024

Part I: Questions

1. What is the difference between an object and a class in OOP**?** **// 1.5M**

Feature	Class	Object
Definition	**Blueprint** or template for creating objects	**Instance** of a class
Structure	Contains fields (attributes) and methods (functions)	Contains actual values for attributes defined by its class
Usage	Encapsulates data and functionalities; allows code reuse and modular design	Interacts with data and methods defined in the class
Memory Allocation	Does not allocate memory for data	Allocates memory to hold actual data
Reusability	Can be reused to create multiple objects	Each object is a unique instance

2. What does encapsulation achieve in OOP? **//01M**
   - Groups data and methods together.
   - Hides data implementation details.
   - Controls access to data through methods (getters/setters).
   - Protects data integrity.
   - Promotes modular and secure code.
3. What is the significance of the ***final*** keyword in Java? **//1.5M**
   - Makes variables constant (e.g., final PI = 3.14159).
   - Prevents methods from being overridden in subclasses.
   - Makes classes final (not inheritable).
4. What is the primary purpose of a constructor in a class? //**01M**

   The primary purpose of a constructor in a class is to initialize an object when it's created
5. Can a Java class have multiple constructors? //01M

   A Java class can absolutely have multiple constructors. These constructors can have different **numbers** or **types** of **parameters**
6. Describe a scenario where the this keyword is necessary in a constructor or method. **//01M**
   Provide a code example **//01M**

   this is often necessary in **constructors** or **methods** to disambiguate between instance variables (**fields**) and **parameters** or **local variables** that have the **same name**.

```
public class Person {
    private String name;
   private int age;
    public Person(String name, int age)   {
            this.name = name;
           this.age = age;
 } //……..
```

Person : 2M **Member** : 2M **Author** : 2M **Book** : 02M **Library** : 03.75M 0.25 :Text Formatting

```
public class Person {                                   //0.5 M
    private String name;
    private String firstName;
    private int age;

    public Person(String name, String firstName, int age) {     // 01M
        this.name = name;
        this.firstName = firstName;
        this.age = age;
    }

    public void display() {                                                 // 0.5 M
        System.out.println("Name: " + name + ", First Name: " + firstName + ", Age: " + age);
    }
}

/*****************************************************************************************************************/

public class Member extends Person {                    // 0.25 M
    private String memberNumber;

    public Member(String name, String firstName, int age, String memberNumber) {    // 0.5 M
        super(name, firstName, age);
        this.memberNumber = memberNumber;
    }

    @Override                                                           // 1.25M
    public void display() {
        super.display();   0.25M
        System.out.println("Member Number: " + memberNumber);
    }
}
```

```
/*****************************************************************************************************************/
public class Author extends Person {                    // 0.25 M
    private String authorNumber;

    public Author(String name, String firstName, int age, String authorNumber) {        // 0.5 M
        super(name, firstName, age);
        this.authorNumber = authorNumber;
    }

    @Override                                                                            // 1.25M
    public void display() {
        super.display();
        System.out.println("Author Number: " + authorNumber);
    }
}
/*****************************************************************************************************************/
public class Book {                                     //0.5M
    private String ISBN;
    private String title;
    private Author[] authors;

    public Book(String ISBN, String title, Author[] authors) {      //0.5M
        this.ISBN = ISBN;
        this.title = title;
        this.authors = authors;
    }

    public void display() {                                         //01M
        System.out.println("ISBN: " + ISBN + ", Title: " + title);
        System.out.println("Authors:");
        for (Author author : authors) {
            author.display();
        }
    }
}
```

```
public class Library {                                  //0.5M
    public static void main(String[] args) {

        // Instantiate a Member
        Member member = new Member("Smith", "John", 25, "M001");        //0.5M

        // Instantiate
        Author author1 = new Author("Doe", "Jane", 45, "A001");          // 0.25M
        Author author2 = new Author("Brown", "Emily", 38, "A002");          // 0.25M

        // Instantiate a Book with two Authors
        Book book = new Book("1234567890", "OOP in Java", new Author[]{author1, author2}); // 0.25M

        // Display the details of the Member, Authors, and the Book
        System.out.println("Member Details:");                   //02M
        member.display();
        System.out.println("\nBook Details:");
        book.display();
    }
}
```

//.25 Text Formatting