University Center Abdelhafid Boussouf - Mila
Institute of Mathematics and Computer Science
2nd Year Computer Science Bachelor's Program

Academic Year: 2024-2025
Subject: Object-Oriented Programming
Tutorial/ Practical Series No. 2

Exercise 1: packages and static methods

The following code contains errors:

```
package first;
class A {
    private int q;
    public static void f(int n) {
        q = n;
    }
    public int getQ() {
        return q;
    }
}
```

```
public class TestA {
    public static void main(String args[]) {
        A a = new A();
        int x = 5;
        System.out.println("Value of q is: " + a.getQ());
    }
}
```

Tasks:

1. Identify the syntax errors and correct them.
2. Explain why the errors occur and how your corrections fix them.

Exercise 2: packages and static methods

1. **Create a package named mathutils and define a class Calculator inside it.** The class should have the following **static methods**:
    - `add(int a, int b)`: Returns the sum of two integers.
    - `multiply(int a, int b)`: Returns the product of two integers.
    - `power(int base, int exponent)`: Returns the result of raising base to the power of exponent.
2. **Create another package named app and define a class MainApp inside it.**
    - In the main method, use the Calculator class to perform the following operations:
        - Add 5 and 10.
        - Multiply 7 and 8.
        - Calculate 2 raised to the power of 5.
    - Print the results of each operation.
3. **Questions:** Can you make the Calculator methods **non static**? What changes would you need to make in the MainApp class?

Exercise 3: Method Overloading and Parameter Passing

1. Create a class **`MathOperations`** with the following overloaded methods:

- int `add(int a, int b)`: Returns the sum of two integers.
- double `add(double a, double b)`: Returns the sum of two decimal numbers.
- `String` `add(String a, String b)`: Returns the concatenation of two strings.
- `int` `add(int a, int b, int c)`: Returns the sum of three integers.

2. Create a class `Param` with the following attributes and methods:

- An attribute `a` of type `int`.
- A constructor that initializes the attribute `a`.
- A method `getA()` that returns the value of `a`.
- `void incrementer(int n)`: Increments the value of *n* by 1
- `void incrementer(Param pm)`: Increments the attribute a of the pm object by 1.

3. Create a class `Test` with a `main` method to test the functionality:

- Declare an integer `x` and initialize it with the value `9`.
- Create an object `p` of the `Param` class and initialize its attribute `a` with the value of `x`.
- Test the `add` methods of the `MathOperations` class with different parameter combinations.
- Test the incrementer methods with the variable `x` and the object `p`.
- Display the values of `x` and the attribute `a` of the object `p` after each method call.

4. Questions

1. Why does the value of **x** not change after calling **incrementer(int n)**?

2. Why does the value of **p.a** change after calling **incrementer(Param pm)**?

3. Can you add another overloaded **add(int a, int b)** that return double ?

Exercise 4: Static Attributes and Arrays

1. Define a class named `Student` that includes the following components:

Attributes:

- `id` (*int*): An attribute that uniquely identifies each student. It must be automatically incremented each time a new student object is created.
- `name` (*String*): The student's last name.
- `firstName` (*String*): The student's first name.
- `age` (*int*): The student's age.
- `score` (*double*): The student's academic score, ranging from 0 to 20.

Methods:

- A constructor that initializes all instance attributes and assigns a unique `id` to each student.
- A static method `getTotalStudents()` that returns the total number of student instances created.
- An instance method `displayInfo()` that displays the student information.

2. Implement a class `TestStudent` containing a `main()` method that performs the following operations:
   - Declares an array of five `Student` objects.
   - Initializes the array with the following student data :

Last Name	**First Name**	**Age**	**Score**
Ahmed	Ali	20	14.5
Benslimane	Aicha	19	12.2
Benbrahim	Idir	22	11.0
Maouche	Lina	21	14.5
Bouzid	Karim	23	15.8

   - Iterates through the array and displays the information of each student.
   - Displays the total number of students created.