Java Institute For Advanced Technology

Department of Examinations

COURSE(S) – (LEADING TO)	PCJT SOFTWARE ENGINEER I
UNIT NAME	OBJECT ORIENTED PROGRAMMING CONCEPTS I
UNIT ID	H7DY 04
ASSIGNMENT ID	H7DY 04/AS/01
OBJECTIVE	DEVELOP A JAVA PROGRAM BASED ON THE GIVEN SCENARIO BY FOLLOWING THE STEP-BY-STEP GUIDE. IMPLEMENT THE REQUIRED FUNCTIONALITY AND SUBMIT A PROJECT AND VIDEO PRESENTATION EXPLAINING YOUR CODE.

IMPORTANT GUIDELINES FOR CANDIDATES

- Maximum video duration: 10 minutes
- The video must be presented in English.
- Use the Zoom platform to record the video (Students must turn on their camera, ensuring a clear view of their face during the demonstration).
- Student Introduction: State your name and Student Portal registered NIC number. The video must include a thumbnail image containing the student's name, NIC, photo, subject, and assignment details (OOPC I Assignment ID). (Duration: 30 seconds)
- Project Explanation and Demonstration: Run the project in NetBeans while demonstrating. The explanation must include Object-Oriented Programming Concepts used in your code.
- File Naming Conventions:
    - Rename the PDF file as: OOPC_I_ASDOC-Your NIC-Your Name
    - Rename the video file as: OOPC_I_ASVID-Your NIC-Your Name
- Submission: Submit both the PDF and video presentation to the student portal.
- More Information: Follow the Assignment Guideline Video

Scenario:
You are designing a **simple combat simulation game** where a soldier can carry up to three different guns. Each gun type has a unique firing mechanism, and the soldier should be able to **switch between guns**, **shoot**, **reload**, and **manage their weapons effectively**. Your task is to implement this simulation using Java, ensuring proper OOP principles.

Assignment Instructions:

- **Create a Soldier and Assign Guns:**
  - Instantiate a **Soldier** object with a name.
  - Create instances of **AK47**, **Pistol**, and **Sniper**, then assign them bullets.
  - Assign these guns to the soldier using an Array.

- **Simulate Actions:**
  - Make the soldier shoot with different guns.
  - Check the behavior when bullets run out.
  - Implement a feature where the soldier can switch between guns.
  - Make the soldier drop and pick guns dynamically.

- **Enhance the Program:**
  - Add a method in Soldier to switch guns.
  - Ensure that the program does not crash if a soldier tries to shoot without a gun.
  - Print clear messages indicating actions performed.

- **Special Requirements:**
  - The **Gun** class should be designed in a way that prevents direct instantiation.
  - Each gun type must define its own shooting behavior.
  - The **Soldier** should have the ability to manage multiple guns dynamically.
  - The **shoot()** method should handle cases where bullets are insufficient.
  - Ensure switching between guns does not cause unexpected exceptions.
  - A soldier can hold a maximum of three guns at a time.