Day 7: Assignment - Serialization, Exceptions & JUnit

Declare following object beans..

→ Person class (id, name, age)

→ Studnet extending Person class (roll_no, marks, no-args constr, param constr) - should implement Serializable interface

→ Employee extending Person class(salary, dept, work_location, no-args constr, param constr) - should implement Externalizable interface so that only to save salary, dept state while serializing

→ PartTimeExmployee (salaryPerHour, consultant_code) - This should not be able to be serialized. If you try to serialize this object, it should throw NotSerializableException

→ SerializeTest class with main() method → From the main method create the collection of Student and collection of Employee and try to serialize them in file - myobjects

→ DeserializeTest with main() method - > Read the serialized file and try to de-serialize the object and observe the state of de-serialized object vs before serialization

Important - Write test-cases for SeializeTest and DeserializeTest classes using JUnit 5

NOTE: The above problem statement examines the following topics

- How serialization work in Java
- What is difference between serializable and Externalizable interfaces
- Object & JVM behavior in case of inheritance and serialization
- Serializing & Deserializing objects with collections

- How to prevent subclass objects from serialization/deserialization
- Use of JUnit framework for writing test cases

Implement following Use-case with respect to Exception Handling

Write a java program with main method.. try to read user input through the scanner class and do some action which should throw user defined checked exception - SomeSeriousProblemOccuredException

```
public class SomeSeriousProblemOccuredException extends Exception{

}

public class SimpleProgram{
  public static void main(String...args){
    Scanner scanner = new Scanner(System.in);
    //some statements throwing SomeSeriousProblemOccuredException;
    scanner.close();
  }
}
```

→ Add try-catch-finally blocks (move scanner.close in finally) in above program's ver1 run to observe the output

→ Now in ver2 of the same program, try to remove finally and catch block with only try.

Hint - use java 7 introduced try-with-resource to auto-close resources like scanner

Interview Questions

Ques 1: What is serialization? When we use it try to explain with any real use-case.

Ques 2: Can you serialize an object of subclass if superclass does not implement serializable

Ques 3: Can you prevent serializing a subclass object when super type is serializable

Ques 4: What is the user of transient keyword in Java explain with use-case

Ques 5: Why serialVersionUID is important in class definitions. When it becomes important to define in objects. What is JVM behavior in case you do not define serialVersionUID for your object

Ques 6: Can we write try block without catch and finally

Quest 7: Can we write try with finally block but not catch block

Ques 8: How to create user defined unchecked exceptions in java

Ques 9: What happens when method returns something from try block, will finally be executed in that case?

Ques 10: Name some important Checked Exceptions you have used in your implementations