**Howard University**

**College of Engineering and Architecture**

**Department of Electrical Engineering & Computer Science**

**Large Scale Programming**

**Fall 2023**

**Prof. Bernard P. Woolfolk**

**First Exam – 100 pts**

**October 19, 2023**

**Name :Johan Lingani**

**Section 1: True/False. Clearly mark your answer. Each question is 1pt.**

**(15 pts.)**

**F** Java allows a class to extend from multiple classes.

**T**  A subclass can add behavior that is not present in the superclass.  
**F** Java arrays can have a variable length, allowing you to resize them dynamically.

**F** if Engine extends CarPart, then a variable of type Engine can be assigned a CarPart object.

**T** If Engine extends CarPart, then a variable of type CarPart can be assigned an Engine object.

**T** Java supports multiple inheritance of classes through interfaces.

**F** Java supports multiple constructors with the same signature within a class.

**F** Java code is a compiled language that creates binary code for a specific architecture and must be recompiled when moved to another machine.

**T** A checked exception represents an error that a program cannot ignore (must either catch or declare).

**F** Good object-oriented design promotes maximizing the use of global variable and making little use of encapsulation.

**T** Deep inheritance hierarchies should be avoided because they can lead to design complexities and maintenance challenges.

**F** Exceptions in Java should be used for error handling and regular control flow.

**T**  Amazon S3 provides an object storage service that allows you to store and retrieve any amount of data at any time.

**F** Amazon EC2 (Elastic Compute Cloud) is a managed service provide by AWS that allows you to run serverless applications. **F** Amazon RDS supports both relational and non-relational databases.

**Section 2: Multiple-Choice. Clearly mark your answer. Each question is 1pt.**

**(20 pts)**

Standard code libraries in Java are organized into

* Interfaces
* Classes
* Objects
* **Packages**

One of the following is NOT TRUE about Object-Oriented Paradigms (OOPs):

* OOP is a set of techniques and processes focusing on how to analyze and model a real-world problem and to design a solution.
* The intended benefits of OOP are to solve the “software” crisis, break complexity into small manageable chunks, and make software maintenance easier.
* OOP allows reuse of components – plug and play
* **OOP solves the entire problem in one program.**

Which point is FALSE from the following?

* **A class is an object**
* A class is a template or prototype that defines the composition and the behavior of all objects of certain kinds.
* A class may have fields of composite types
* From a class you may instantiate/create an object.

A process that involves recognizing and focusing on the important characteristics of a situation or object is known as:

* Encapsulation
* Polymorphism
* **Abstraction**
* Inheritance
* Object persistence

An object-oriented design tends to focus on blank to identify objects:

* Verbs
* **Nouns**
* Encapsulation
* Inheritance

Which of the following is one of the benefits of object-oriented programming?

* Enables code reuse.
* More understandable code.
* More maintainable code.
* **All of the above.**

Which of the following characteristics of an object-oriented programming language restricts behavior so that an object can only perform actions that are defined for its class?

* Dynamic Binding
* Polymorphism
* Inheritance
* **Encapsulation**

In Java, if there are one or more constructors for a class then

* **Exactly one of the constructors will be called each time an object of that class is created**
* All of the constructors will be called each time an object of that class is created
* A destructor must also be written.
* None of the above, classes cannot have constructors

What is true about “has-a” and “is-a” relationships? **(Choose two)**.

* **Instance variables can be used when creating a has-a relationship.**
* **Inheritance represents an is-a relationship.**
* Inheritance represents a has-a relationship.
* Instances must be used when creating a has-a relationship.

Which of the following is true about a method inherited into a class C?

* It must be defined in C before C can be instantiated
* It must be overridden in C before C can be instantiated
* It always forces C to become abstract
* **It overrides any method in C with the same name**

The static variable declared in a class is called

* Global variable
* Local Variable
* **Class variable**
* Instance variable

Which feature of OOP is exhibited by the function overriding?

* **Polymorphism**
* Encapsulation
* Abstraction
* Inheritance

Which of the following is an application of the principle of inheritance?

a) An object of class A has a reference to a class B object.

b) Several methods have the same name, but have different signatures.

c) Objects created with new are allocated on the heap.

d) Fields are usually declared private.

**e) All classes are ultimately derived from the super class called Object**

The term signature can be used when describing a method. In this context, a signature is:

a) Defined by the import statement for that class.

**b) The number of arguments and the data type of each argument.**

c) The return type of method (for example, double or String).

d) The variable names in the argument list.

e) All of the above

In the example code fragment shown below, the keyword **abstract**:

**public abstract class Test { // . . . more class code }**

* **Implies that no object of type Test can ever be**

**created.**

* Makes class Test independent of all other classes, in

particular, it is not a subclass of the class Object.

* Requires the implementation of the toString() method.
* Ensures that only one object of type Test is ever created.
* All of the above.

Which of the following statements about constructors are correct:

* A constructor has the same name as the class name.
* A constructor is responsible for the initialization of an objects’ instance fields.
* Constructor methods have no return type.
* A class can have several constructors.
* **All of the above.**

Which of these can be overloaded? (1pt.)

* methods
* constructors
* **both a & b**
* classes
* Interfaces

Why would a class be declared as abstract?

* **Because it doesn’t make logical sense to instantiate it**
* So that it can be used as an interface
* So that it cannot be inherited from
* Because it has no abstract

Given the code. What is true?

public class Room {

private int roomNr;

private Date beginDtm;

private Date endDttm;

public void book(int roomNr, Date beginDttm, Date endDttm) {

this.roomNr = roomNr;  
 this.beginDtm = beginDttm;  
 this.endDttm = endDttm;

}

}

* the code demonstrates polymorphism
* **the class is fully encapsulated**
* the variable roomNr breaks encapsulation
* variables beginDttm and endDttm break polymorphism
* the method book breaks encapsulation

A single try block must be followed by which of these?

* finally
* catch
* **catch or finally**
* catch and finally

**Section #3: Short answers. Express your answer clearly but don’t write a dissertation either.**

**(20 pts.)**

Describe the principles of encapsulation and abstraction in object-oriented programming. Discuss how these principles contribute to build more maintainable and modular code.

* Encapsulation- the process of creating data “capsules/bundles” composed of variables and methods which are accessible only through public and private “gates”. This is useful to control certain data points or methods to be accessed.
* Abstraction- the process of hiding unnecessary complexity of objects while highlighting the necessary features.  
  This is like a car dashboard with the steering wheel, gas, and brake to allow the driver to interact with the car.

Discuss the difference between method overloading and method overriding in Java. Clearly explain when you would use each technique.

* Overloading: This is when you have multiple methods in the same class with the same name but different parameters. It's like having different flavors of the same drink.
* Overriding: This is when a subclass provides a new version of a method that is already defined in its superclass. It's like getting the family recipe for a dish but adding your own twist.  
    
  When to Use:
* Overloading: When you want the same operation to be performed in different ways depending on the input.
* Overriding: When the child class needs to give a specific implementation to a method already defined in its parent class.

A program reads a web address and tries to connect to the site. As the application developer of the Java class and method that receives the connection message, would you implement the method to throw a checked or unchecked exception when a connection cannot be established? Please defend your answer.

* I would use a checked exception. They need to be explicitly caught. In this scenario connecting to the site seems like a crucial thing.

Briefly describe the difference between a class and an interface in Java.

* Classes are like blueprints for objects. They contain all the properties and attributes of said object
* Interface: An interface is more like a contract that classes can agree to implement. It declares what methods a class must have, but it doesn't specify how those methods should be implemented.