Course Title: Big Java, Early Objects

Chapter Number: 12 Object-Oriented Design

Question type: Multiple Choice

1) Which of the following most likely indicates that you have chosen a good name for your class?

a) The name consists of a single word.

b) You can tell by the name what an object of the class is supposed to represent.

c) You can easily determine by the name how many concepts the class represents.

d) The name describes what task the class will perform.

Answer: b

Title: Which indicates a good class name?

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

2) Which of the following questions should you ask yourself in order to determine if you have named your class properly?

a) Does the class name contain 8 or fewer characters?

b) Is the class name a verb?

c) Can I visualize an object of the class?

d) Does the class name describe the tasks that this class will accomplish?

Answer: c

Title: How do I ensure I have named a class properly?

Difficulty: Easy

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

3) Which of the following would be an appropriate name for a game-related class?

a) FinalizeLevelOne

b) InitialLevel

c) ResetTheCurrentLevel

d) AscendToFinalLevel

Answer: b

Title: Which would be an appropriate class name?

Difficulty: Medium

.

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

4) Which of the following would be an appropriate name for a class used in an inventory application?

a) Product

b) InitializeInventoryLevel

c) ResetCurrentInventoryLevel

d) ReceiveNewProducts

Answer: a

Title: Which would be an appropriate class name?

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

5) You are designing an application to support an automobile rental agency. Which of the following probably should NOT be represented as an object?

a) Auto

b) Customer

c) Payment amount

d) Rental contract

Answer: c

Title: Which entity probably should NOT be represented as an object?

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

6) You are designing an application to support a veterinary clinic. Which of the following probably should NOT be represented as an object?

a) Pet

b) Customer

c) Medical service performed

d) Drug dosage

Answer: d

Title: Which entity probably should NOT be represented as an object?

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

7) Classes often correspond to \_\_\_\_ in a requirements description.

a) Verbs

b) Nouns

c) Dependencies

d) Relationships

Answer: b

Title: Classes correspond to \_\_\_\_ in a requirements description.

Difficulty: Easy

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

8) A/an \_\_\_\_ represents a set of objects with the same behavior.

a) Association

b) Aggregation

c) Dependency

d) Class

Answer: d

Title: A/an \_\_\_\_ represents a set of objects with the same behavior.

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.1 Discovering classes

9) After you have identified a set of classes needed for a program, you should now \_\_\_\_.

a) Define the behavior of each class.

b) Look for nouns that describe the tasks.

c) Begin writing the code for the classes.

d) Establish the relationships between the classes.

Answer: a

Title: After identifying the classes needed for a program, you should now \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.2 The CRC Card Method

10) A CRC card describes \_\_\_\_.

a) A class, its responsibilities, and its collaborating classes.

b) A class, its methods, and its constructors.

c) A class, its relationships, and its collaborating classes.

d) A class, its requirements, and its concurrent classes.

Answer: a

Title: A CRC card describes \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.2 The CRC Card Method

11) When using the CRC method, other classes that are needed to fulfill the responsibilities of a given class are called its \_\_\_\_.

a) Related classes.

b) Responsible classes.

c) Collaborators.

d) Concurrent classes.

Answer: c

Title: When using the CRC method, other classes that are needed are called \_\_\_.

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.2 The CRC Card Method

12) When using the CRC method, if you determine that a class cannot carry out a particular task by itself and needs the help of another class to complete the task, you should \_\_\_\_.

a) Do nothing on this class’s CRC card. The task will be shown on the other class’s CRC card.

b) Add this class onto the other class’s CRC card as a collaborator.

c) Add the other class onto this class’s CRC card as a collaborator.

d) Add each class onto the other class's CRC card as a collaborator.

Answer: c

Title: When using the CRC method, if a class needs help from another class to complete a task

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.2 The CRC Card Method

13) Which of the following is the most important consideration when designing a class?

a) Each class should represent an appropriate mathematical concept.

b) Each class should represent a single concept or object from the problem domain.

c) Each class should represent no more than three specific concepts.

d) Each class should represent multiple concepts only if they are closely related.

Answer: b

Title: What is the most important consideration when designing a class?

Difficulty: Medium

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.3 Cohesion

14) Under which of the following conditions would the public interface of a class be considered cohesive?

a) All of its features are public and none of its features are static.

b) The quality of the public interface is rated as moderate to high.

c) All of its features are related to the concept that the class represents.

d) It is obvious that the public interface refers to multiple concepts.

Answer: c

Title: When is the public interface of a class considered cohesive?

Difficulty: Hard

Section Reference 1: 12.1 Classes and Their Responsibilities

Section Reference 2: 12.1.3 Cohesion

15) A class (ClassOne) is considered to have a dependency on another class (ClassTwo) under which of the following conditions?

a) Each class uses objects of the other.

b) The public interfaces of both classes are cohesive.

c) ClassTwo uses objects of ClassOne.

d) ClassOne uses objects of ClassTwo.

Answer: d

Title: Under what conditions does one class have a dependency on another?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

16) A UML class diagram would be most useful in visually illustrating which of the following?

a) the cohesiveness of a class’s interface.

b) the amount of complexity of a class’s interface.

c) dependencies between classes.

d) relationships between classes and their interfaces.

Answer: c

Title: For what are UML class diagrams most useful?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

17) Why is it generally considered good practice to minimize coupling between classes?

a) Low coupling increases the operational efficiency of a program.

b) High coupling implies less interface cohesion.

c) High coupling increases program maintenance and hinders code reuse.

d) Low coupling decreases the probability of code redundancy.

Answer: c

Title: Why is it considered good practice to minimize class coupling?

Difficulty: Hard

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

18) Dependency between classes denotes that \_\_\_\_.

a) Objects of one class inherit from objects of its superclass.

b) Objects of one class can be inherited by its subclasses.

c) Objects of one class implement an interface.

d) Methods in the objects of one class use an object of another class.

Answer: d

Title: Dependency between classes denotes that \_\_\_\_.

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

19) In a UML diagram, dependency is denoted by \_\_\_\_.

a) A dotted/dashed line with a closed arrow tip.

b) A solid line with a closed arrow tip.

c) A dotted/dashed line with an open arrow tip.

d) A solid line with an open arrow tip.

Answer: c

Title: In a UML diagram, dependency is denoted by \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

20) If many classes of a program depend on each other, we say that \_\_\_\_.

a) cohesiveness is high.

b) cohesiveness is low.

c) coupling is high.

d) coupling is low.

Answer: c

Title: If many classes of a program depend on each other, we say that \_\_\_\_.

Difficulty: Easy

.

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

21) UML means\_\_\_\_.

a) Unified Mode Language

b) User Modeling Language

c) User Mode Language.

d) Unified Modeling Language.

Answer: d

Title: UML means\_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

22) The dependency relationship is sometimes referred to as the \_\_\_\_ relationship.

a) is-a

b) has-a

c) depends-on

d) knows-about

Answer: d

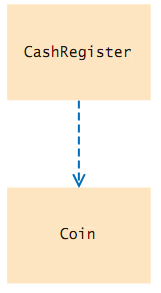
Title: The dependency relationship is sometimes referred to as the \_\_\_ relationship.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

23) Which statement correctly describes the class relationship shown in this diagram?



a) CashRegister class depends on Coin class

b) Coin class depends on CashRegister class

c) CashRegister class aggregates Coin class.

d) Coin class inherits from CashRegister class.

Answer: a

Title: Which statement describes the class relationship?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

24) Which of the following code snippets denotes that the Purse class depends on the Wallet class?

a)

public class Purse extends Wallet

b)

public class Wallet extends Purse

c)

public class Wallet

{

private Purse aPurse;

}

d)

public class Purse

{

private Wallet aWallet;

}

Answer: d

Title: Which code snippet denotes dependency?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

25) Which of the following code snippets denotes that the Pen class depends on the Ink class?

a)

public class Pen extends Ink

b)

public class Ink extends Pen

c)

public class Pen

{

private Ink anInk;

}

d)

public class Ink

{

private Pen aPen;

}

Answer: c

Title: Which code snippet denotes dependency?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.1 Dependency

26) Which of the following code snippets denotes that the Kitchen class depends on the Stove class?

a)

public class Kitchen extends Stove

b)

public class Stove

{

private Kitchen[] kitchens;

}

c)

public class Kitchen

{

private Stove aStove;

}

d)

public class Kitchen implements Stove

Answer: c

Title: Which code snippet denotes dependency?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

27) \_\_\_\_ is often described as the *has-a* relationship.

a) Inheritance.

b) Aggregation.

c) Polymorphism.

d) Dependency.

Answer: b

Title: \_\_\_\_ is often described as the *has-a* relationship.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

28) Which of the following statements is correct?

a) Dependency is a stronger form of aggregation.

b) Aggregation indicates high cohesiveness.

c) Aggregation indicates low coupling.

d) Aggregation is a stronger form of dependency.

Answer: d

Title: Which of the following statements is correct?

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

29) Consider the following code snippet:

public class Motorcycle

{

private Tire[] tires;

. . .

}

This code is best described as an example of \_\_\_\_.

a) Inheritance.

b) Aggregation.

c) Polymorphism.

d) Association.

Answer: b

Title: This code is an example of \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

30) Consider the following code snippet:

public class Motorcycle

{

private Tire[] tires;

. . .

}

Which of the following statements correctly describes the relationship between the Motorcycle and Tire classes?

a) Motorcycle exhibits the *has-a* relationship with regards to Tire.

b) Motorcycle exhibits the *is-a* relationship with regard to Tire.

c) Tire exhibits the *has-a* relationship with regards to Motorcycle.

d) Tire exhibits the *is-a* relationship with regards to Motorcycle.

Answer: a

Title: Which statement describes the relationship between these classes?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

31) Consider the following code snippet:

public class Purse

{

private Coin[] coins;

. . .

}

This code is best described as an example of \_\_\_\_.

a) Inheritance.

b) Aggregation.

c) Polymorphism.

d) Association.

Answer: b

Title: This code is an example of \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

32) Consider the following code snippet:

public class Purse

{

private Coin[] coins;

. . .

}

Which of the following statements is correct?

a) Purse aggregates Coin.

b) Coin depends on Purse.

c) The Coin class must be declared as an inner class within Purse.

d) Coin inherits from Purse.

Answer: a

Title: Which statement about this code is correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

33) Aggregation denotes that \_\_\_\_.

a) Objects of one class inherit from objects of its superclass.

b) Objects of one class can be inherited by its subclasses.

c) Objects of one class contain references to objects of its superclass.

d) Objects of one class contain references to objects of another class.

Answer: d

Title: Aggregation denotes that \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

34) A CashRegister class contains an array list of Coin objects. This is best described as an example of \_\_\_\_.

a) Association

b) Inheritance

c) Cohesiveness

d) Aggregation

Answer: d

Title: This is an example of \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

35) A Quiz class contains an array of Question objects. This is best described as an example of \_\_\_\_.

a) Aggregation

b) Cohesiveness

c) Association

d) Inheritance

Answer: a

Title: This relationship is an example of \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

36) In a UML diagram, aggregation is denoted by \_\_\_\_.

a) A solid line with a diamond-shaped symbol next to the aggregating class.

b) An arrow with an open triangle pointing to the aggregating class.

c) A dotted line with a diamond-shaped symbol next to the aggregating class.

d) A dotted line with an open arrow tip pointing to the aggregating class.

Answer: a

Title: In a UML diagram, aggregation is denoted by \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

37) In general, you need \_\_\_\_ when an object needs to remember another object between method calls.

a) inheritance

b) aggregation

c) association

d) an interface implementation

Answer: b

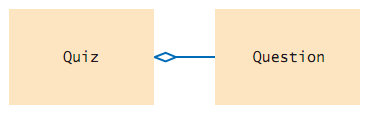
Title: In general, you need \_\_\_\_ when an object needs to remember another object.

Difficulty: Hard

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

38) Which statement correctly describes the class relationship shown in this diagram?



a) Quiz class depends on Question class

b) Question class aggregates Quiz class

c) Quiz class aggregates Question class.

d) Question class inherits from Quiz class.

Answer: c

Title: Which statement describes the class relationship?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

39) Which of the following code snippets denotes that the Fleet class aggregates the Taxi class?

a)

public class Fleet extends Taxi

b)

public class Taxi extends Fleet

c)

public class Fleet

{

private ArrayList<Taxi> taxicabs;

}

d)

public class Taxi

{

private Fleet myFleet;

}

Answer: c

Title: Which code snippet denotes aggregation?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

40) Which of the following code snippets denotes that the Kitchen class aggregates the Dish class?

a)

public class Kitchen extends Dish

b)

public class Dish extends Kitchen

c)

public class Kitchen

{

private Dish[] dishes;

}

d)

public class Dish

{

private Kitchen myKitchen;

}

Answer: c

Title: Which code snippet denotes aggregation?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.2 Aggregation

41) \_\_\_\_ is often described as the *is-a* relationship.

a) Inheritance.

b) Aggregation.

c) Polymorphism.

d) Dependency.

Answer: a

Title: \_\_\_\_ is often described as the *is-a* relationship.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

42) Consider the following code snippet:

public class Motorcycle extends Vehicle

{

private Tire[] tires;

. . .

}

This code is best described as an example of \_\_\_\_.

a) Inheritance and multiplicities.

b) Aggregation and dependency.

c) Inheritance and aggregation.

d) Inheritance and association.

Answer: c

Title: This code is an example of \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

43) Consider the following code snippet:

public class Motorcycle extends Vehicle

{ . . . }

Which of the following statements correctly describes the relationship between the Motorcycle and Vehicle classes?

a) Motorcycle exhibits the *has-a* relationship with regards to Vehicle.

b) Motorcycle exhibits the *is-a* relationship with regard to Vehicle.

c) Vehicle exhibits the *has-a* relationship with regards to Motorcycle.

d) Vehicle exhibits the *is-a* relationship with regards to Motorcycle.

Answer: b

Title: Which statement describes the relationship between these classes?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

44) Consider the following code snippet:

public class Motorcycle extends Vehicle

{

private Tire[] tires;

private Engine anEngine;

. . .

}

Which of the following statements describing relationships between the Motorcycle, Tire, Engine and Vehicle classes is NOT correct?

a) Motorcycle exhibits the *has-a* relationship with regards to Tire.

b) Motorcycle exhibits the *is-a* relationship with regard to Vehicle.

c) Vehicle exhibits the *has-a* relationship with regards to Motorcycle.

d) Motorcycle exhibits the *knows-about* relationship with regards to Engine.

Answer: c

Title: Which statement about the relationship between these classes is NOT correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

45) Consider the following code snippet:

public class PowerBoat extends Vessel

{

private Engine[] engines;

. . .

}

This code is best described as an example of \_\_\_\_.

a) Inheritance and multiplicities.

b) Aggregation and association.

c) Inheritance and aggregation.

d) Inheritance and dependency.

Answer: c

Title: This code is an example of \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

46) Consider the following code snippet:

public class SailBoat extends Vessel

{

private Engine[] engines;

private Sail mainsail;

. . .

}

Which of the following statements is NOT correct?

a) SailBoat inherits from Vessel.

b) Sail depends on Sailboat.

c) Sailboat aggregates Engine.

d) SailBoat aggregates Sail.

Answer: b

Title: Which statement about this code is NOT correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

47) Consider the following code snippet:

public class SailBoat extends Vessel

{ . . . }

public class Catamaran extends SailBoat

{ . . . }

Which of the following statements is NOT correct?

a) SailBoat inherits from Vessel.

b) Catamaran inherits from Vessel.

c) Catamaran inherits from Sailboat.

d) Catamaran depends on SailBoat.

Answer: d

Title: Which statement about this code is NOT correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

48) Consider the following code snippet:

public class Manager extends Employee

{

private Project[] projects;

private Address address;

. . .

}

Which of the following statements is NOT correct?

a) Manager inherits from Employee.

b) Address aggregates Manager.

c) Manager aggregates Address.

d) Manager aggregates Project.

Answer: b

Title: Which statement about this code is NOT correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

49) In a UML diagram, inheritance is denoted by \_\_\_\_.

a) A solid line with a diamond-shaped symbol next to the superclass.

b) A solid line with a closed arrow tip pointing to the superclass.

c) A dotted line with an open arrow tip pointing to the superclass.

d) A solid line with an open arrow tip pointing to the superclass.

Answer: b

Title: In a UML diagram, inheritance is denoted by \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

50) Which of the followings statements about class relationships is correct?

a) Inheritance represents the *is-a* relationship, while aggregation represents the *has-a* relationship.

b) Inheritance represents the *has-a* relationship, while dependency represents the *uses* relationship.

c) Aggregation represents the *is-a* relationship, while inheritance represents the *uses* relationship.

d) Aggregation represents the *has-a* relationship, while dependence represents the *is-a* relationship.

Answer: a

Title: Which statement about class relationships is correct?

Difficulty: Hard

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

51) Which of the following code snippets denotes that the Lime class inherits from the Citrus class?

a)

public class Lime extends Citrus

b)

public class Citrus extends Lime

c)

public class Lime

{

private ArrayList<Citrus> fruits;

}

d)

public class Citrus

{

private ArrayList<Lime> limes;

}

Answer: a

Title: Which code snippet denotes inheritance?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

52) Which of the following code snippets denotes that the Rose class inherits from the Flower class?

a)

public class Flower extends Rose

b)

public class Rose extends Flower

c)

public class Rose

{

private Flower[] flowers;

}

d)

public class Flower

{

private Rose[] limes;

}

Answer: b

Title: Which code snippet denotes inheritance?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

53) When designing classes, if you find classes with common behavior you should \_\_\_\_.

a) Combine them all into a single class.

b) Place the common behavior into a superclass.

c) Place the common behavior into a subclass.

d) Give them similar names.

Answer: b

Title: When designing classes with common behavior, you should \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

54) In a UML diagram, an interface implementation is denoted by \_\_\_\_.

a) A dotted line with a closed arrow tip.

b) A solid line with a closed arrow tip.

c) A dotted line with an open arrow tip.

d) A solid line with an open arrow tip.

Answer: a

Title: In a UML diagram, an interface implementation is denoted by \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

55) In a UML diagram, the relationship symbol shown below denotes \_\_\_\_.



a) inheritance

b) aggregation

c) dependency

d) interface implementation

Answer: b

Title: In a UML diagram, the relationship symbol shown denotes \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

56) In a UML diagram, the relationship symbol shown below denotes \_\_\_\_.



a) inheritance

b) aggregation

c) dependency

d) interface implementation

Answer: d

Title: In a UML diagram, the relationship symbol shown denotes \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

57) In a UML diagram, the relationship symbol shown below denotes \_\_\_\_.



a) inheritance

b) aggregation

c) dependency

d) interface implementation

Answer: a

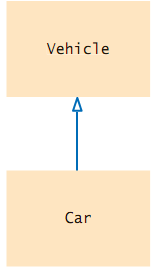
Title: In a UML diagram, the relationship symbol shown denotes \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

58) Which statement correctly describes the class relationship shown in this diagram?



a) Vehicle class depends on Car class

b) Car class depends on Vehicle class

c) Vehicle class inherits from Car class.

d) Car class inherits from Vehicle class.

Answer: d

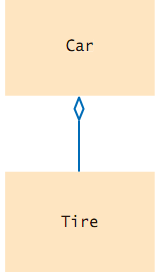
Title: Which statement describes the class relationship shown?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

59) Which statement correctly describes the class relationship shown in this diagram?



a) Tire class aggregates Car class.

b) Car class aggregates Tire class.

c) Car class inherits from Tire class.

d) Tire class inherits from Car class.

Answer: b

Title: Which statement describes the class relationship shown?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

60) You have determined a need for a Book class and a Page class in your program. Which relationship is most appropriate between these classes?

a) Inheritance

b) Aggregation

c) Dependency

d) Interface implementation

Answer: b

Title: Which relationship is most appropriate between these classes?

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

61) You have determined a need for an Employee class and a TemporaryEmployee class in your program. Which relationship is most appropriate between these classes?

a) Inheritance

b) Aggregation

c) Dependency

d) Interface implementation

Answer: a

Title: Which relationship is most appropriate between these classes?

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: 12.2.3 Inheritance

62) \_\_\_\_ relationships come from the collaboration columns on the CRC cards.

a) Association

b) Inheritance

c) Dependency

d) Attribute

Answer: c

Title: \_\_\_\_ relationships come from the collaboration columns on the CRC cards.

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: How To 12.1 CRC cards and UML diagrams

63) When using CRC cards, UML diagrams should be created \_\_\_.

a) Prior to discovering classes.

b) After you have discovered classes and their relationships.

c) During the implementation phase.

d) To produce final documentation of the system.

Answer: b

Title: When using CRC cards, UML diagrams should be created \_\_\_.

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: How To 12.1 CRC cards and UML diagrams

64) Which of the following statements about class attributes is true?

a) An attribute always corresponds to an instance variable of the class.

b) An attribute is an externally observable property that objects of the class have.

c) An attribute is used to represent aggregation with another class.

d) Attributes cannot be represented in UML diagrams.

Answer: b

Title: Which statement about class attributes is true?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Special Topic 12.1 Attributes and Methods in UML Diagrams

65) How does a UML diagram denote classes and their attributes and methods?

a) A class rectangle contains the class name in the top, methods in the middle, and attributes in the bottom.

b) A class rectangle contains the class name in the top, attributes in the middle, and methods in the bottom.

c) A class rectangle contains the methods in the top, class name in the middle, and attributes in the bottom.

d) A class rectangle contains the attributes in the top, methods in the middle, and class name in the bottom.

Answer: b

Title: How does a UML diagram denote classes and their attributes and methods?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Special Topic 12.1 Attributes and Methods in UML Diagrams

66) How does a UML diagram denote a multiplicity of one or more in an aggregation relationship?

a) \*

b) 1..\*

c) \*..1

d) 1..n

Answer: b

Title: How does a UML diagram denote a multiplicity of one or more?

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Special Topic 12.2 Multiplicities

67) Given the following diagram:



What does this diagram indicate about the relationship between the customers and bank accounts?

a) A bank account may be owned by only one customer.

b) A bank account may be owned by one or more customers.

c) A customer may have only one bank account.

d) A customer may have one or more bank accounts.

Answer: d

Title: What does this diagram indicate about the relationship between the customers and bank accounts?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Special Topic 12.2 Multiplicities

68) Which of the following statements about associations between classes is true?

a) A class is associated with another class if both classes are in the same package.

b) A class is associated with another class if the class inherits from the other class.

c) A class is associated with another class if the other class inherits from this class.

d) A class is associated with another class if you can navigate from objects of one class to objects of the other class.

Answer: d

Title: Which statement about associations between classes is correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Special Topic 12.3 Aggregation, Association, and Composition

69) You have determined the need for a File class and a Folder class in your program. Which of the following would best describe the relationship between these classes?

a) Aggregation

b) Association

c) Composition

d) Inheritance

Answer: c

Title: Which statement about relationships between classes is correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Special Topic 12.3 Aggregation, Association, and Composition

70) If you have parallel arrays or array lists of the same length which each store a part of what could be considered an object, \_\_\_\_.

a) You should rewrite the program to use a two-dimensional array.

b) You should rewrite the program to use a two-dimensional array list.

c) You should create a class to hold the related data in the ith slice of each array, and then create an arraylist of objects.

d) There are no alternatives to this type of program design.

Answer: c

Title: Which statement about parallel arrays or array lists is correct?

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Programming Tip 12.1 Make Parallel Arrays into Arrays of Objects

71) Parallel arrays are \_\_\_\_.

a) Two or more arrays or array lists of the same length which each store a part of what could be considered an object.

b) Two or more arrays or array lists of the same length.

c) Two or more arrays or array lists of the same data type.

d) Two or more arrays or array lists that are declared one after the other in a program.

Answer: a

Title: Parallel arrays are \_\_\_\_

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Programming Tip 12.1 Make Parallel Arrays into Arrays of Objects

72) To improve the quality of the public interface of a class, \_\_\_\_.

a) You should maximize cohesion and minimize inheritance.

b) You should minimize cohesion and maximize coupling.

c) You should maximize cohesion and minimize coupling.

d) You should minimize cohesion and maximize aggregation.

Answer: c

Title: How to improve the quality of the public interface of a class

Difficulty: Medium

Section Reference 1: 12.2 Relationships Between Classes

Section Reference 2: Programming Tip 12.2 Consistency

73) Select a code segment to complete the Name class, so that it reflects a dependency relationship between Name and String.

public class Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends String

{

…

}

b)

{

private String first;

private String second;

…

}

c)

implements String

{

…

}

d)

{

private ArrayList<String> names;

…

}

Answer: b

Title: Select code segment that reflects dependency relationship

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

74) Select a code segment to complete the Team class, so that it reflects an aggregation relationship between Team and Player.

public class Team \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends Player

{

…

}

b)

{

private Player aPlayer;

…

}

c)

implements Player

{

…

}

d)

{

private ArrayList<Player> players;

…

}

Answer: d

Title: Select code segment that reflects aggregation relationship

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

75) Select a code segment to complete the Player class, so that it reflects an inheritance relationship between Player and Person.

public class Player \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends Person

{

…

}

b)

{

private Person thePlayer;

…

}

c)

implements Person

{

…

}

d)

{

private ArrayList<Person> players;

…

}

Answer: a

Title: Select code segment that reflects inheritance relationship

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

76) Select a code segment to complete the SmartPhone class, so that it reflects an interface implementation relationship between SmartPhone and MP3Player.

public class SmartPhone \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends MP3Player

{

…

}

b)

{

private MP3Player musicPlayer;

…

}

c)

implements MP3Player

{

…

}

d)

implements interface MP3Player

{

…

}

Answer: c

Title: Select code segment that reflects interface implementation relationship

Difficulty: Easy

Section Reference 1: 12.2 Relationships Between Classes

77) The textbook recommends a five-part program development process consisting of the following activities:

I Use UML diagrams to record class relationships.

II Gather requirements.

III Implement the program.

IV Use CRC cards to identify classes.

V Use javadoc to document method behavior.

Which is the correct order in which these activities should be completed?

a) IV, II, I, III, V

b) IV, II, I, V, III

c) II, IV, I, III, V

d) II, IV, I, V, III

Answer: d

Title: Recognize the five-part development process

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.1 Requirements

78) Before you begin designing a solution, you should \_\_\_\_.

a) Document the methods that will be needed.

b) Gather and document all requirements for the program in plain English.

c) Determine the classes that will be needed.

d) Create the UML diagrams to represent the classes.

Answer: b

Title: Before you begin designing a solution, you should \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.1 Requirements

79) You are designing a software solution for an automobile rental company. You have decided that the following nouns apply to the requirements: Auto, Customer, Address, Rental Contract, Mileage, Rental Date, Daily Rate, Total. Which of these should be represented as classes?

a) All of them.

b) Auto, Customer, Address, Rental Contract.

c) Auto, Customer, Address, Rental Contract, Mileage.

d) Auto, Customer, Address, Rental Contract, Mileage, Rental Date, Daily Rate.

Answer: b

Title: Which of these should be represented as classes?

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.2 CRC Cards

80) You are designing a software solution for an automobile rental company. You have decided that the following nouns apply to the requirements: Auto, Customer, Address, Rental Contract, Mileage, Rental Date, Daily Rate, Total. Which of these should be represented as instance variables?

a) Mileage

b) Mileage, Daily Rate.

c) Mileage, Rental Date, Daily Rate.

d) Mileage, Rental Date, Daily Rate, Total.

Answer: c

Title: Which of these should be represented as instance variables?

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.2 CRC Cards

81) You are designing a software solution for a veterinary clinic. The clinic provides various services for each pet on each visit. You have decided that the following nouns apply to the requirements: Customer, Address, Pet, Visit, Visit Date, Service Charge, Total Charge, Next Appointment. Which of these should be represented as classes?

a) All of them.

b) Customer, Address, Pet, Visit.

c) Customer, Address, Pet, Visit, Service Charge.

d) Customer, Address, Pet, Visit, Service Charge, Total Charge, Visit Date, Next Appointment.

Answer: b

Title: Which of these should be represented as classes?

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.2 CRC Cards

82) You are designing a software solution for a veterinary clinic. The clinic provides various services for each pet on each visit. You have decided that the following nouns apply to the requirements: Customer, Address, Pet, Visit, Visit Date, Service Charge, Total Charge, Next Appointment. Which of these should be represented as instance variables?

a) Service Charge

b) Service Charge, Visit Date.

c) Service Charge, Visit Date, Next Appointment.

d) Service Charge, Visit Date, Next Appointment, Total Charge.

Answer: c

Title: Which of these should be represented as instance variables?

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.2 CRC Cards

83) You are designing a software solution for an automobile rental company. A customer may rent only a single auto at a given time. You have decided that the following classes are needed: Auto, Customer, Address, Rental Contract. Which of these should be represented as aggregation?

a) Rental Contract class aggregates Address class

b) Rental Contract class aggregates Customer and Address classes.

c) Rental Contract class aggregates Customer, Address, and Auto classes.

d) Rental Contract class does not aggregate any of the other classes.

Answer: c

Title: Which of these should be represented as aggregation?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.3 UML Diagrams

84) You are designing a software solution for a veterinary clinic. The clinic provides various services for each pet on each visit. You have decided that the following classes are needed: Customer, Address, Pet, and Visit. Which of these should be represented as aggregation?

a) Customer class aggregates Address, Pet, and Visit classes.

b) Visit class aggregates Address class.

c) Pet class aggregates Address class.

d) Pet class aggregates Customer class.

Answer: a

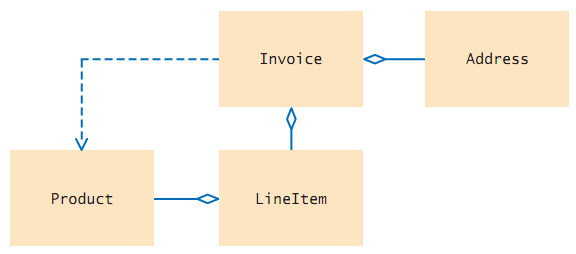
Title: Which of these class relationships should be represented as aggregation?

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.3 UML Diagrams

85) Given the following diagram showing class relationships:



What type of relationship is shown between Invoice and LineItem?

a) LineItem inherits from Invoice.

b) LineItem aggregates Invoice.

c) Invoice aggregates LineItem.

d) Invoice inherits from LineItem.

Answer: c

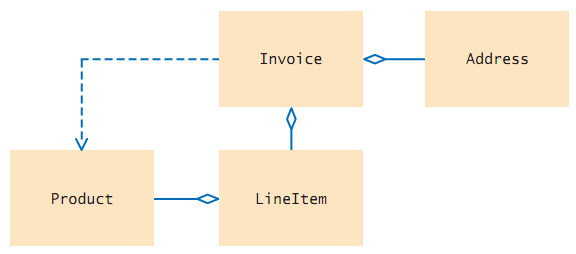
Title: What type of relationship is shown in this diagram?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.3 UML Diagrams

86) Given the following diagram showing class relationships:



What type of relationship is shown between Invoice and Product?

a) Product inherits from Invoice.

b) Product aggregates Invoice.

c) Invoice inherits from Product.

d) Invoice is dependent on Product.

Answer: d

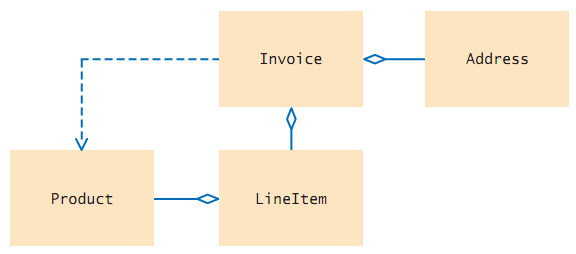
Title: What type of relationship is shown in this diagram?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.3 UML Diagrams

87) Given the following diagram showing class relationships:



What type of relationship is shown between LineItem and Product?

a) Product inherits from LineItem.

b) Product aggregates LineItem.

c) LineItem aggregates Product.

d) LineItem inherits from Product.

Answer: c

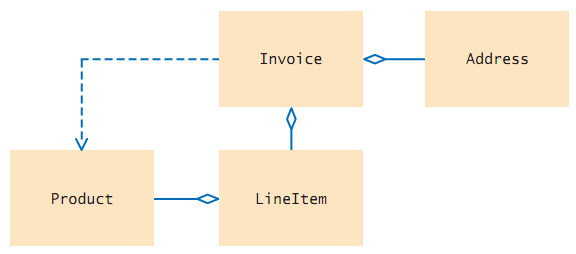
Title: What type of relationship is shown in this diagram?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.3 UML Diagrams

88) Given the following diagram showing class relationships:



What type of relationship is shown between Invoice and Address?

a) Address inherits from Invoice.

b) Invoice aggregates Address.

c) Address aggregates Invoice.

d) Invoice inherits from Address.

Answer: b

Title: What type of relationship is shown in this diagram?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.3 UML Diagrams

89) The final step of the design phase recommended by the textbook is to \_\_\_\_.

a) Write the documentation of the discovered classes and methods

b) Write the code for each class and all of its methods.

c) Identify the classes and methods that will be required.

d) Create the UML diagrams to represent the classes.

Answer: a

Title: The final step of the design phase is to \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.4 Method Documentation

90) When documenting discovered classes and methods during the design phase, you should \_\_\_\_.

a) Create a Java source file for each class, write the method signatures, and use javadoc comments to describe the methods, but leave the method bodies blank.

b) Create a Java source file for each class and write code to implement all of its methods, and use javadoc comments to document them.

c) Create a Java source file for each class, use javadoc comments to document the methods that are to be added later, but do not write any code for the methods.

d) Documentation of methods should not be done during the design phase.

Answer: a

Title: When documenting discovered classes and methods during the design phase, you should \_\_\_\_.

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.4 Method Documentation

91) Which of the following can be used to record the behavior of classes?

a) Javadoc comments

b) Nouns and verbs in the problem description

c) Polymorphism

d) UML notation

Answer: a

Title: Which can be used to record the behavior of classes?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.4 Method Documentation

92) During the implementation phase, which of the following statements is true?

a) Aggregated classes will yield subclasses.

b) Aggregated classes will yield instance variables.

c) Aggregated classes will yield class packages.

d) Aggregated classes will yield interfaces.

Answer: b

Title: During the implementation phase, which statement is true?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.5 Implementation

93) During the implementation phase, which of the following statements is true?

a) Method parameter variables are available from the CRC cards.

b) Method parameter variables are available from the javadocs created during the design phase.

c) You must determine the method parameter variables during implementation phase.

d) Method parameter variables are available from the UML diagrams.

Answer: b

Title: During the implementation phase, which statement method parameter variables is true?

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.5 Implementation

94) During the implementation phase, which of the following statements is true?

a) Class instance variables are available from the CRC cards.

b) Class instance variables are available from the javadocs created during the design phase.

c) You must determine the class instance variables during implementation phase.

d) Class instance variables can be determined from the UML diagrams.

Answer: d

Title: During the implementation phase, which statement about class instance variables is true?

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: 12.3.5 Implementation

95) When using UML to create state diagrams, a state is denoted by \_\_\_\_.

a) A rectangle with double border.

b) A rectangle with rounded corners.

c) An oval with double border.

d) A standard rectangle with square corners.

Answer: b

Title: In a UML state diagram, a state is denoted by \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: Worked Example 12.1 Simulating an Automatic Teller Machine

96) When using UML to create state diagrams, state change is denoted by \_\_\_\_.

a) Arrows with labels that connect rectangles.

b) Double arrows with labels that connect rectangles.

c) An oval between two rectangles, connected with arrows.

d) Dividing the state rectangle into several parts, one for each state change.

Answer: a

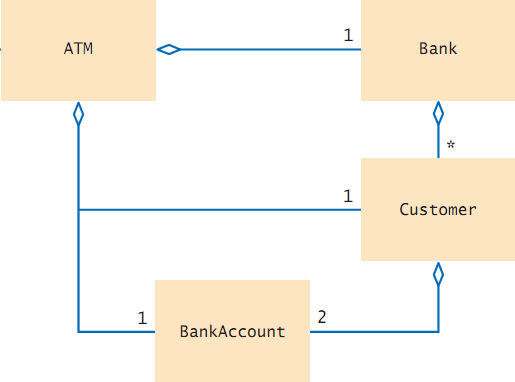
Title: In a UML state diagram, a state change is denoted by \_\_\_\_.

Difficulty: Easy

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: Worked Example 12.1 Simulating an Automatic Teller Machine

97) Given the following diagram showing class relationships:



What of the following best describes the type of relationship shown between Bank and Customer?

a) Customer aggregates Bank, indicating that a customer may deal with many banks.

b) Bank aggregates Customer, indicating that a bank may deal with many customers.

c) Bank depends on Customer, indicating that a bank may deal with many customers.

d) Customer depends on Bank, indicating that a customer may deal with many banks.

Answer: b

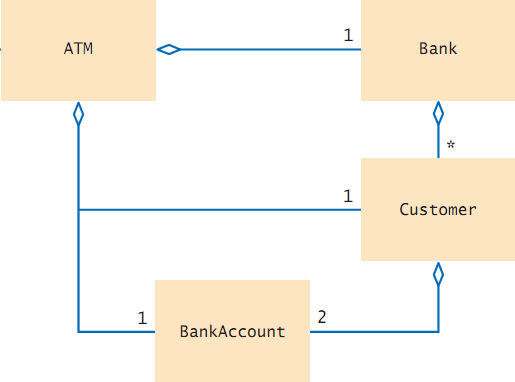
Title: What type of relationship is shown in this diagram?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: Worked Example 12.1 Simulating an Automatic Teller Machine

98) Given the following diagram showing class relationships:



What type of relationship is shown between BankAccount and Customer?

a) Customer aggregates BankAccount, indicating that a customer may have 2 bank accounts.

b) BankAccount aggregates Customer, indicating that 2 customers may share a bank account.

c) BankAccount depends on Customer, indicating that many customers may have 2 bank accounts.

d) Customer depends on BankAccount, indicating that a customer may have 2 bank accounts.

Answer: a

Title: What type of relationship is shown in this diagram?

Difficulty: Hard

Section Reference 1: 12.3 Application: Printing an Invoice

Section Reference 2: Worked Example 12.1 Simulating an Automatic Teller Machine

99) Suppose that the invoice-printing application from section 12.3 needs to be enhanced by including the computation and printing of sales taxes as needed. Since some vendors charge no sales taxes, the original Invoice class needs to be preserved. Select the code segment that best illustrates reuse of the existing Invoice class.

public class TaxableInvoice \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends Invoice

{

…

}

b)

{

private Invoice originalInvoice;

…

}

c)

implements Invoice

{

…

}

d)

{

private ArrayList<Invoice> invoices;

…

}

Answer: a

Title: Select code segment that illustrate reuse of existing Invoice class

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

100) Regarding the invoice-printing application from section 12.3, it is decided that a Customer class is needed to store customer information, including the name, billing address, and a customer id number to be printed at the top of each invoice. Since the existing Address class already stores the customer’s name and physical address, the new Customer class can simply reuse Address. Select the code segment that best illustrates how the Customer class can reuse the existing Address class.

public class Customer \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends Address

{

private String idNumber;

…

}

b)

{

private Address information;

private String idNumber;

…

}

c)

implements Address

{

private String idNumber;

…

}

d)

{

private Address information;

private String name;

private String street;

private String city;

private String state;

private String zip;

private String idNumber;

…

}

Answer: b

Title: Select code segment that illustrates reuse of existing Address class

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

101) Suppose that the invoice-printing application from section 12.3 needs to be enhanced by making it possible for class InvoicePrinter to store several invoices to be printed. Select the code segment that best illustrates reuse of the existing Invoice class for this purpose.

public class InvoicePrinter \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends Invoice

{

…

}

b)

{

public static void main(String[] args)

{

Invoice invoice1;

Invoice invoice2;

…

}

c)

implements Invoice

{

…

}

d)

{

public static void main(String[] args)

{

ArrayList<Invoice> invoices;

…

}

}

Answer: d

Title: Select code segment that illustrate reuse of existing Invoice class to store several invoices

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

102) Suppose you are developing a payroll application that computes and displays weekly paycheck amounts for various employees. As a result of the design phase, the partial Employee class below is developed. Select the method header that best completes the class, according to the method comments.

public class Employee

{

private int hoursWorked;

/\*\*

Computes the weekly salary for this employee.

@param hourlyRate the rate per hour earned by the employee

@return the weekly salary

\*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

{

// method body

}

}

a) public void computeSalary(double hourlyRate)

b) public double computeSalary(double hourlyRate, int hoursWorked)

c) public double computeSalary(double hourlyRate)

d) public double computeSalary(int hoursWorked)

Answer: c

Title: Select method header that corresponds to class design and documentation

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

103) Suppose you are developing a payroll application that computes and displays weekly paycheck amounts for various employees. As a result of the design phase, an Employee class is developed. In addition, the Payroll class is designed to store and process information for various employees. Select the code segment that best completes the Payroll class.

public class Payroll \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a)

extends Employee

{

…

}

b)

{

public static void main(String[] args)

{

Employee anEmployee;

…

}

c)

{

public static void main(String[] args)

{

int hoursWorked;

double hourlyRate;

…

}

}

d)

{

public static void main(String[] args)

{

ArrayList<Employee> employees;

…

}

}

Answer: d

Title: Select code segment that illustrate reuse of existing Employee class to store several employees

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice

104) Suppose you are developing a payroll application that computes and displays weekly paycheck amounts for various employees. As a result of the design phase, the partial Employee class below is developed. Select the method header that best completes the class, according to the method comments.

public class Employee

{

private double hourlyRate;

private int hoursWorked;

/\*\*

Modifies the hourly rate for this employee.

@param newHourlyRate the rate per hour earned by the employee

\*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

{

// method body

}

}

a) public void changeRate(double newHourlyRate)

b) public void changeRate(double newHourlyRate, int hoursWorked)

c) public double changeRate(double newHourlyRate)

d) public double changeRate(double newHourlyRate, int hoursWorked)

Answer: a

Title: Select method header that corresponds to class design and documentation

Difficulty: Medium

Section Reference 1: 12.3 Application: Printing an Invoice