44-542 Object Oriented Programming

Exam 2 Version B KEY (1.25 points each, 100 points total)

* Answer all the questions
* Shade the correct option in the given scantron sheet with pencil
* For some questions you need to refer to given figure

1. An abstract method must be overridden by all subclasses, except for subclasses also declared as abstract.
   1. True
   2. False
2. An abstract class cannot be instantiated.
   1. True
   2. False
3. A class that contains an abstract method must be declared as abstract.
   1. True
   2. False
4. In Java, it is possible that an abstract class can have no abstract methods.
   1. True
   2. False
5. In Java, a class may be both a superclass and a subclass. That is, given classes A, B, and C, it is possible that A is a subclass of B and a superclass of C.
   1. True
   2. False
6. A class can extend multiple classes.
   1. True
   2. False
7. A class can implement multiple interfaces.
   1. True
   2. False
8. A class can extend one or more interfaces.
   1. True
   2. False
9. An interface can extend multiple interfaces.
   1. True
   2. False
10. List interface extends \_\_\_\_\_\_\_\_\_\_\_\_\_
    1. Collection class
    2. Collection interface
    3. Both a and b
11. The keyword static in method main means the JVM does not have to create an instance of the class in order to start the program.
    1. True
    2. False
12. The \_\_\_\_\_\_\_\_\_ keyword in method main provides the visibility to enable the JVM to start the program.
    1. public
    2. void
    3. static
13. The \_\_\_\_\_\_\_\_\_ keyword in method main means the program does not return data to the JVM when it ends.
    1. public
    2. void
    3. static
14. Use accompanying figure to determine how the following are related: MinorAccount, ChildrenAccount
    1. MinorAccount extends ChildrenAccount
    2. ChildrenAccount extends MinorAccount
    3. MinorAccount implements ChildrenAccount
    4. ChildrenAccount implements MinorAccount
15. Use accompanying figure to determine how the following are related: Account, MajorAccount
    1. Account implements MajorAccount
    2. MajorAccount implements Account
    3. MajorAccount extends Account
    4. Account extends MajorAccount
16. Use accompanying figure to determine how the following are related: AbstractAccount, MajorAccount
    1. AbstractAccount abstracts MajorAccount
    2. AbstractAccount implements MajorAccount
    3. MajorAccount abstracts AbstractAccount
    4. MajorAccount implements AbstractAccount
17. Use accompanying figure to determine how the following are related: ChildrenAccount, KidsAccount
    1. ChildrenAccount extends KidsAccount
    2. KidsAccount extends ChildrenAccount
    3. ChildrenAccount implements KidsAccount
    4. KidsAccount implements ChildrenAccount
18. Use accompanying figure to determine legality: Account a1 = new Account();
    1. Legal
    2. Illegal
19. Use accompanying figure to determine legality: SavingsAccount sa1 = new SavingsAccount();
    1. Legal
    2. Illegal
20. Use accompanying figure to determine legality: MajorAccount ma1 = new MajorAccount();
    1. Legal
    2. Illegal
21. Use accompanying figure to determine legality: AbstractAccount aa1 = new AbstractAccount();
    1. Legal
    2. Illegal
22. Use accompanying figure to determine legality: SavingsAccount sa2 = new PersonalSavingsAccount();
    1. Legal
    2. Ilegal
23. Use accompanying figure to determine legality: AbstractAccount aa2 = new SavingsAccount();
    1. Legal
    2. Illegal
24. Use accompanying figure to determine legality: CheckingAccount ca1 = new SavingsAccount();
    1. Legal
    2. Illegal
25. Use accompanying figure to determine legality: CreditAccount credit1 = new FamilySavingsAccount();
    1. Legal
    2. Illegal
26. Use accompanying figure to determine legality: ChildrenAccount ca2 = new KidsAccount();
    1. Legal
    2. Illegal
27. Use accompanying figure to determine legality: Account a2 = new MajorAccount();
    1. Legal
    2. Illegal
28. Use accompanying figure to determine legality: MajorAccount ma3 = new AbstractAccount();
    1. Legal
    2. Illegal
29. Which of the following is a correct option (use accompanying figure)?
    1. A MajorAccount is a MinorAccount
    2. A MinorAccount is a MajorAccount
    3. An AbstractAccount is a MajorAccount
    4. An Abstract Account is a CheckingAccount
30. Use the accompanying figure to determine whether this statement is true: A PersonalSavingsAccount is a SavingsAccount
    1. True
    2. False
31. Use the accompanying figure to determine whether this statement is true: A FamilySavingsAccount is a MajorAccount
    1. True
    2. False
32. Use the accompanying figure to answer this question: Which of the following is true?
    1. A KidsAccount is a Account
    2. A CreditAccount is a AbstractAccount
    3. Both a and b are correct
33. Using the accompanying figure, consider the following code segment.

Line 1) SavingsAccount mySavAcct1 = new FamilySavingsAccount();

Line 2) System.out.println(mySavAcct1.findInterestRate());

* Assume that the constructor statement is correct; that is, FamilySavingsAccount has a no-arg constructor.
* Assume both SavingsAccount and FamilySavingsAccount have method findInterestRate.

When the code in Line 2 is executed, Java will invoke the findInterestRate method from FamilySavingsAccount. This is an example of

* 1. inheritance
  2. polymorphic substitution
  3. late-binding polymorphism

1. Using the accompanying figure, consider the following code segment.

Line 1) FamilySavingsAccount fSave1 = new FamilySavingsAccount();

Line 2) SavingsAccount save1 = new SavingsAccount();

Line 3) fSave1 = save1;

Line 4) save1 = fSave1;

Assume that the constructor statements are correct; that is, both FamilySavingsAccount and SavingsAccount have no-arg constructors.

Which of the above lines is a correct example of polymorphic substitution?

* 1. Line 1
  2. Line 2
  3. Line 3
  4. Line 4

1. Using the accompanying figure, consider the following code segment.

Line 1) FamilySavingsAccount acct1 = new FamilySavingsAccount();

Line 2) System.out.println(acct1.findInterestRate());

* Assume that the constructor statement is correct; that is, FamilySavingsAccount has a constructor with no parameters.
* Assume SavingsAccount has a method findInterestRate, but FamilySavingsAccount does not have such a method.

Line 2 is correct and is an example of

* 1. inheritance
  2. polymorphic substitution
  3. late-binding polymorphism

1. In Java 8, which of the following is true of interfaces?
   1. An interface must have at least one default method.
   2. An interface cannot have any default methods.
   3. An interface may have any number of default methods.
2. Using the accompanying figure, consider the following code segment.

FamilySavingsAccount acct1 = new FamilySavingsAccount();

SavingsAccount acct2 = new SavingsAccount();

acct1 = (FamilySavingsAccount) acct2;

Which of the following is true of this code segment?

* 1. This code will not compile; there is a syntax error.
  2. This code will compile, but a runtime exception will occur.
  3. This code will compile and run.

1. Using the accompanying figure, consider the following code segment.

KidsAccount acct1 = new KidsAccount();

ChildrenAccount acct2 = new KidsAccount();

acct1 = (KidsAccount) acct2;

Which of the following is true of this code segment?

* 1. This code will not compile; there is a syntax error.
  2. This code will compile, but a runtime exception will occur.
  3. This code will compile and run.

1. Using the accompanying figure, consider the following code segment.

KidsAccount acct1 = new KidsAccount();

ChildrenAccount acct2 = new KidsAccount();

acct2 = acct1;

Which of the following is true of this code segment?

* 1. This code will not compile; there is a syntax error.
  2. This code will compile, but a runtime exception will occur.
  3. This code will compile and run.

1. Assume fSave1, and cat1 are the references of FamilySavingsAccount, and Cat classes respectively. Which is true of the following Java statement?

fsave1 = (FamilySavingsAccount) cat1;

* 1. A ClassCastException will be thrown.
  2. A compilation error will occur.
  3. A CastMismatchExecution error will occur.

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ testing shows that individual pieces are working correctly.
   1. Black-box
   2. White-box
   3. Unit
2. In \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ testing, tester does not need to know the program implementation.
   1. Black-box
   2. White-box
   3. Unit
3. In \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ testing, test data is selected on the basis of program implementation.
   1. Black-box
   2. White-box
   3. Unit
4. Which of the following is correct in choosing the input values for testing?
   1. Check values inside and outside of the acceptable range
   2. Check boundary values
   3. Both a and b
5. Java provides a framework for testing called \_\_\_\_\_\_\_\_\_\_\_
   1. JavaTest
   2. JUnit
   3. TestDriver
6. Annotations can affect how tools and libraries interact with a program.
   1. True
   2. False
7. The general-purpose annotations can be used to provide metadata (data about the data).
   1. True
   2. False
8. cities is a name of an ArrayList of String objects. Which of the following gives the size of the array list?
   1. cities.length
   2. cities.length()
   3. cities.size()
9. Which of the following is an incorrect annotation?
   1. @return
   2. @param
   3. @overload
10. \_\_\_\_\_\_\_\_\_\_\_\_\_ means the superclass and subclass have a method with same signature.
    1. Overloading
    2. Overriding
11. The “is-a test” says that
    1. an instance of the superclass has an instance of the subclass as a private instance variable
    2. an instance of the subclass has an instance of the superclass as a private instance variable
    3. an instance of the superclass is an instance of the subclass
    4. an instance of the subclass is an instance of the superclass
12. When we add new instance variables or methods to a subclass, the superclass
    1. has access to the new instance variables and methods
    2. has no access to the new instance variables and methods
    3. has access to the new instance variables, but not to the new methods
    4. has access to the new methods, but not to the new instance variables
13. What is the visibility indicator of private attributes in UML?
    1. +
    2. –
    3. #
14. In Java, casting of objects is possible.
    1. True
    2. False
15. An ArrayList is fixed size and cannot grow.
    1. True
    2. False
16. ArrayLists only store object references.
    1. True
    2. False
17. In order to use an ArrayList you must import\_\_\_\_\_\_\_\_\_\_\_\_
    1. java.lang.ArrayList
    2. java.util.ArrayList
    3. java.collection.ArrayList
18. In an ArrayList, an exception is thrown if you attempt to add an element at index *i* if *i* is greater than the current size of the list.
    1. True
    2. False
19. In ArrayLists, the no-arg constructor creates an empty list with capacity of \_\_\_\_\_\_\_\_\_ elements.
    1. 10
    2. 100
    3. Infinity
20. The size of the ArrayList is not necessarily the same as the capacity of the ArrayList.
    1. True
    2. False
21. We can leave empty spaces in an ArrayList.
    1. True
    2. False
22. Which of the following is a wrapper class?
    1. Math
    2. Int
    3. Integer
23. Which of the following methods is use to add an element at the end of the ArrayList?
    1. add(e element)
    2. add(i index, e element)
    3. both a and b will work
24. ArrayList is a generic class and can be used with different types.
    1. True
    2. False
25. In a UML diagram, if no visibility indicator is used with an attribute, then the visibility of the attribute is \_\_\_\_\_\_\_\_\_\_\_\_\_
    1. public
    2. private
    3. package
26. For an attribute, how many visibility choices are there?
    1. 2
    2. 3
    3. 4
27. When we extend the state or behavior of a class, the original class is called\_\_\_\_\_\_
    1. super class
    2. sub class
    3. semi super class
28. The class that extends the state or behavior of the original class is called \_\_\_\_\_\_\_\_\_\_\_
    1. super class
    2. sub class
    3. semi super class
29. The sub class inherits the following from the super class.
    1. Attributes
    2. Methods
    3. Both a and b
30. In order for a class to extend another class, it must pass the \_\_\_\_\_\_\_\_ test.
    1. has-a
    2. is-a
    3. with-a
31. Which of the following is incorrect about protected variables?
    1. Can be accessed by all subclasses
    2. Cannot be accessed by sub classes
    3. Can be accessed by the class itself
32. Sub classes inherit the protected methods of the super class.
    1. True
    2. False
33. Inheritance in Java 8 is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    1. One-way
    2. Two-way
34. Every Java class automatically extends Object class and inherits methods of Object class.
    1. True
    2. False
35. If JVM does not find an implementation of toString() until it gets to the Object class, then it uses the method provided by \_\_\_\_\_\_\_\_\_\_\_\_.
    1. Object class
    2. Super class (not the Object class)
    3. Sub class
36. The constructor in a subclass can invoke the constructor of the superclass using which keyword?
    1. extends
    2. super
    3. implements
37. The ability to override methods coupled with the run-time determination of which method to invoke is referred to as *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*
    1. Polymorphic substitution
    2. Late binding polymorphism
    3. Middle binding polymorphism
38. The fact that a reference variable for the supertype may actually store a reference to an instance of the subclass refers to \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    1. Polymorphic substitution
    2. Late binding polymorphism
    3. Middle binding polymorphism
39. A subclass may override a method originally defined in the superclass by supplying its own definition of the method.
    1. True
    2. False
40. An abstract method has a method body.
    1. True
    2. False