1. Which answer choice can replace line 6 so the code continues to produce the same output?

List<String> rug = new ArrayList<>();

rug.add("circle");

rug.add("square");

System.out.println(rug);

A. System.out.println(rug.asString);

B. System.out.println(rug.asString());

C. System.out.println(rug.toString);

D. System.out.println(rug.toString());

2. Which best describes this code?

class Stats {

private int data;

public int getData() {

return data;

}

public void setData(int data) {

this.data = data;

}

}

A. It is a singleton.

B. It is well encapsulated.

C. It is immutable.

D. It is both well encapsulated and immutable.

3. What design pattern or principle ensures that there will be no more than one instance

of a class?

A. Encapsulation

B. Immutability

C. Singleton

D. Static

4. What is the output of this code?

class Laptop extends Computer {

public void startup() {

System.out.print("laptop-");

}

}public class Computer {

public void startup() {

System.out.print("computer-");

}

public static void main(String[] args) {

Computer computer = new Laptop();

Laptop laptop = new Laptop();

computer.startup();

laptop.startup();

}

}

A. computer-laptop-

B. laptop-computer-

C. laptop-laptop-

D. None of the above

5. Which method can be correctly inserted into this class to meet the contract of the equals() method? You may assume that text is not null.

class Button {

private String text;

public int hashCode() {

return text.hashCode();

}

}

A.

public boolean equals(Object o) {

if ( o == null ) return true;

if (! (o instanceof Button)) return false;

return text.equals(o.text);

}

B.

public boolean equals(Object o) {

if ( o == null ) return true;

Button b = (Button) o;

return text.equals(b.text);

}

C.

public boolean equals(Object o) {

if (! (o instanceof Button)) return false;

return text.equals(o.text);

}

D.

public boolean equals(Object o) {

if (! (o instanceof Button)) return false;

Button b = (Button) o;

return text.equals(b.text);

}

6. Fill in the blanks: \_\_\_\_\_\_\_\_\_\_\_\_means the state of an object cannot be changed

while \_\_\_\_\_\_\_\_\_\_\_\_means that it can.

A. Immutability, mutability

B. Rigidity, flexibility

C. Static, instance

D. None of the above

7. Which is the first line to fail to compile?

class Tool {

void use() { } // r1

}

class Hammer extends Tool {

private void use() { } // r2

public void bang() { } // r3

}

A. r1

B. r2

C. r3

D. None of the above

8. Which of these classes properly implement(s) the singleton pattern?

class ExamAnswers {

private static ExamAnswers instance = new ExamAnswers();

private List<String> answers = new ArrayList<>();

public static List<String> getAnswers() {

return instance.answers;

}

}c

lass TestAnswers {

private static TestAnswers instance = new TestAnswers();

private List<String> answers = new ArrayList<>();

public static TestAnswers getTestAnswers() {

return instance;

}

public List<String> getAnswers() {

return answers;

}

}

A. ExamAnswers

B. TestAnswers

C. Both classes

D. Neither class

9. What does the following print?

public class Transport {

static interface Vehicle {}

static class Bus implements Vehicle {}

public static void main(String[] args) {

Bus bus = new Bus();

boolean n = null instanceof Bus;

boolean v = bus instanceof Vehicle;

boolean b = bus instanceof Bus;

System.out.println(n + " " + v + " " + b);

}

}

A. true true true

B. false true true

C. false false false

D. None of the above

10. Which variable declaration is the first line not to compile?

class Building {}

class House extends Building{}

public void convert() {

Building b = new Building();

House h = new House();

Building bh = new House();

Building p = (House) b;

House q = (Building) h;

Building r = (Building) bh;

House s = (House) bh;

}

A. p

B. q

C. r

D. s

11. Which statement is true about the code that can fill in the blank?

class Sticker {

public int hashCode() {

return 1;

}

public boolean equals(Object o) {

return\_\_\_\_\_\_\_\_\_\_\_\_ ;

}

}

A. It must return false.

B. It must return true.

C. It can return either true or false.

D. None of the above.

12. Which of these classes best implement(s) the singleton pattern?

class ExamAnswers {

private static ExamAnswers instance = new ExamAnswers();

private List<String> answers = new ArrayList<>();

private ExamAnswers() {}

public ExamAnswers getExamAnswers() {

return instance;

}

public List<String> getAnswers() {

return answers;

}

}c

lass TestAnswers {

private static TestAnswers instance = new TestAnswers();

private List<String> answers = new ArrayList<>();

private TestAnswers() {}

public static TestAnswers getTestAnswers() {

return instance;

}

public List<String> getAnswers() {

return answers;

}

}

A. ExamAnswers

B. TestAnswers

C. Both classes

D. Neither class

13. How many lines does the following code output?

public class Cars {

static {

System.out.println("static");

}

private static void drive() {

System.out.println("fast");

}

public static void main(String[] args) {

drive();

drive();

}

}

A. One

B. Two

C. Three

D. None of the above. The code does not compile.

14. What is lazy instantiation?

A. A technique that can be used in an immutable class to wait until the first use to create the object

B. A technique that can be used in a singleton to wait until the first use to create the object

C. A technique that can be used in an immutable class to save memory when creating the object

D. A technique that can be used in a singleton to save memory when creating the object.

15. Which of the following is required for all valid lambda expressions?

A. ()

B. ->

C. {}

D. Parameter data type(s)

16. What is the output of the following application?

package holiday;

enum DaysOff {

Thanksgiving, PresidentsDay, ValentinesDay

}p

ublic class Vacation {

public static void main(String... unused) {

final DaysOff input = DaysOff.Thanksgiving;

switch(input) {

default:

case DaysOff.ValentinesDay:

System.out.print("1");

case DaysOff.PresidentsDay:

System.out.print("2");

}

}

}

A. 1

B. 2

C. 12

D. None of the above

17. Fill in the blanks: A functional interface must contain or inherit \_\_\_\_\_\_\_\_\_\_\_\_and

may optionally include\_\_\_\_\_\_\_\_\_\_\_\_ .

A. at least one abstract method, the @Override annotation

B. exactly one method, static methods

C. exactly one abstract method, the @FunctionalInterface annotation

D. at least one static method, at most one default method

18. Which of the following is a valid lambda expression?

A. r -> {return 1==2}

B. (q) -> true

C. (x,y) -> {int test; return test>0;}

D. a,b -> true

19. Which of the following properties of an enum can be marked abstract?

A. The enum class definition

B. An enum method

C. An enum value

D. None of the above

20. What is the output of the following application?

package world;

public class Matrix {

private int level = 1;

class Deep {

private int level = 2;

class Deeper {

private int level = 5;

public void printReality() {

System.out.print(level);

System.out.print(" "+Matrix.Deep.this.level);

System.out.print(" "+Deep.this.level);

}

}

}

public static void main(String[] bots) {

Matrix.Deep.Deeper simulation = new Matrix().new Deep().new Deeper();

simulation.printReality();

}

}

A. 1 1 2

B. 5 2 2

C. 5 2 1

D. The code does not compile.

21. A local inner class can access which type of local variables?

I. final

II. private

III. effectively final

A. I only

B. I and II

C. III only

D. I and III

22. What is the output of the following application?

package finance;

enum Currency {

DOLLAR, YEN, EURO

}

abstract class Provider {

protected Currency c = Currency.EURO;

}

public class Bank extends Provider {

protected Currency c = Currency.DOLLAR;

public static void main(String[] pennies) {

int value = 0;

switch(new Bank().c) {

case 0:

value--; break;

case 1:

value++; break;

}

System.out.print(value);

}

}

A. 0

B. 1

C. The code does not compile.

D. The code compiles but throws an exception at runtime.

23. What statement best describes the notion of effectively final in Java?

A. A local variable that is marked final

B. A static variable that is marked final

C. A local variable that is not marked final but whose primitive value or object reference does not change after it is initialized

D. A local variable that is not marked final but whose primitive value or object reference does not change after a certain point in the method

24. What is the output of the following application?

package race;

interface Drive {

int SPEED = 5;

default int getSpeed() { return SPEED; }

}interface Hover {

int MAX\_SPEED = 5;

default int getSpeed() { return MAX\_SPEED; }

}

public class Car implements Drive, Hover {

public static void main(String[] gears) {

class RaceCar extends Car {

@Override public int getSpeed() { return 10; }

};

System.out.print(new RaceCar().getSpeed());

}

}

A. 5

B. 10

C. The code does not compile due to the definition of Racecar.

D. The code does not compile for some other reason.

25. Fill in the blanks: It is possible to extend an \_\_\_\_\_\_\_\_\_\_\_\_but not an

\_\_\_\_\_\_\_\_\_\_\_\_.

A. interface, abstract class

B. abstract class, enum

C. enum, interface

D. abstract class, interface

26. Which of the following results is not a possible output of this program?

package sea;

enum Direction { NORTH, SOUTH, EAST, WEST; };

public class Ship {

public static void main(String[] compass) {

System.out.print(Direction.valueOf(compass[0]));

}

}

A. WEST is printed.

B. south is printed.

C. An ArrayIndexOutOfBoundsException is thrown at runtime.

D. An IllegalArgumentException is thrown at runtime.

27. Which of the following is not an advantage of using enumerated types in Java?

A. Ensure consistency of data across an application.

B. Offer ability to create new enumerated values at runtime.

C. Provide access to fixed constants whose value does not change during the course of

the application.

D. Support cases where a value can only take one of a limited number of options.

28. Given the following enum declaration, how many lines contain compilation errors?

package rainbow;

enum Light {}

public enum Color extends Light {

RED, BLUE, ORANGE, GREEN

protected Color() {}

}

A. None, the code compiles as is.

B. One

C. Two

D. Three

29. Which of the following cannot include a static method in its definition?

A. Abstract class

B. Static nested class

C. Interface

D. Local inner class

30. What is the output of the following application?

package ai;

interface Pump {

void pump(double psi);

}i

nterface Bend extends Pump {

void bend(double tensileStrength);

}p

ublic class Robot {

public static final void apply(Bend instruction, double input) { // r1

instruction.bend(input);

}

public static void main(String... future) {

final Robot r = new Robot();

r.apply(x -> System.out.print(x+" bent!"), 5);

}

}

A. 5.0 bent!

B. The code does not compile because Bend is not a functional interface.

C. The code does not compile because of line r1.

D. None of the above.

31. What is the best reason for applying the @Override annotation to a method?

A. It is required to implement an interface method.

B. It is required to override a method.

C. The method will fail to compile if it is not actually overriding another method.

D. There are no good reasons other than as a form of documentation.

32. What is the output of the following application?

package space;

public class Bottle {

public static class Ship {

private enum Sail { // w1

TALL {protected int getHeight() {return 100;}},

SHORT {protected int getHeight() {return 2;}};

protected abstract int getHeight();

}

public Sail getSail() {

return Sail.TALL;

}

}

public static void main(String[] stars) {

Bottle bottle = new Bottle();

Ship q = bottle.new Ship(); // w2

System.out.print(q.getSail());

}

}

A. TALL

B. The code does not compile because of line w1.

C. The code does not compile because of line w2.

D. The code compiles but the application does not produce any output at runtime.

33. Which of the following is not a valid lambda expression?

A. (Integer j, k) -> 5

B. (p,q) -> p+q

C. (Integer x, Integer y) -> x\*y

D. (left,right) -> {return "null";}

34. What is the output of the following application?

1: package fruit;

2:

3: interface Edible { void eat(); }

4: public class ApplePicking {

5: public static void main(String[] food) {

6: Edible apple = new Edible() {

7: @Override

8: void eat() {

9: System.out.print("Yummy!");

10: }

11: }

12: }

13: }

A. The application completes without printing anything.

B. Yummy!

C. One line of this application fails to compile.

D. Two lines of this application fail to compile.

35. What is the output of the following application?

package forest;

public class Woods {

static class Tree {}

public static void main(String[] leaves) {

int water = 10+5;

final class Oak extends Tree { // p1

public int getWater() {

return water; // p2

}

}

System.out.print(new Oak().getWater());

}

}

A. 15

B. The code does not compile because of line p1.

C. The code does not compile because of line p2.

D. None of the above

36. Fill in the blanks: \_\_\_\_\_\_\_\_\_\_\_\_allow Java to support multiple inheritance, and

anonymous inner classes can \_\_\_\_\_\_\_\_\_\_\_\_of them.

A. Abstract classes, extend at most one

B. Abstract classes, extend any number

C. Interfaces, implement at most one

D. Interfaces, implement any number

37. What is the output of the following application?

package vessel;

class Problem extends Exception {}

abstract class Danger {

protected abstract void isDanger() throws Problem;

}

public class SeriousDanger extends Danger {

protected void isDanger() throws Exception { // m1

throw new RuntimeException();

}

public static void main(String[] will) throws Throwable { // m2

new SeriousDanger().isDanger(); // m3

}

}

A. The code does not compile because of line m1

B. The code does not compile because of line m2.

C. The code does not compile because of line m3.

D. The code compiles but throws an exception at runtime.

38. Which of the following is not a true statement about interfaces and abstract classes?

A. Interfaces can only extend other interfaces, while abstract classes can extend both abstract and concrete classes.

B. Unlike abstract classes, interfaces can be marked final.

C. Abstract classes offer support for single inheritance, while interfaces offer support for multiple inheritance.

D. All methods and variables in interfaces are public, while abstract classes can use various access modifiers for their methods and variables, including private in some cases.

39. Which of the following can fill in the blank to make the code compile?

public class News<\_\_\_\_\_\_\_\_> {}

I. ?

II. News

III. Object

A. None of them

B. I

C. II and III

D. I, II, and III

40. Which method is available on both List and Stream implementations?

A. filter()

B. forEach()

C. replace()

D. sort()

41. We are running a library. Patrons select books by name. They get at the back of the checkout line. When they get to the front, they scan the book’s ISBN. The checkout system finds the book based on this number and marks the book as checked out. Of these choices, which data structures best represent the line to check out the book and

the book lookup to mark it as checked out, respectively?

A. ArrayDeque, TreeMap

B. ArrayDeque, TreeSet

C. ArrayList, TreeMap

D. ArrayList, TreeSet

42. Which cannot fill in the blank for this code to compile?

Collection<String> c = new \_\_\_\_\_\_\_\_\_\_\_\_<>();

c.add("pen");

c.remove("pen");

System.out.println(c.isEmpty());

A. ArrayDeque

B. TreeMap

C. TreeSet

D. All of these can fill in the blank.

43. Suppose we want to implement a Comparator<String> so that it sorts the longest strings first. You may assume there are no nulls. Which method could implement such a comparator?

A.

public int compare(String s1, String s2) {

return s1.length() - s2.length();

}

B.

public int compare(String s1, String s2) {

return s2.length() – s1.length();

}

C.

public int compare(Object obj1, object obj2) {

String s1 = (String) obj1;

String s2 = (String) obj2;

return s1.length() - s2.length();

}

D.

public int compare(Object obj1, object obj2) {

String s1 = (String) obj1;

String s2 = (String) obj2;

return s2.length() – s1.length();

}

44. Suppose we want to store JellyBean objects. Which of the following pairs require JellyBean to implement the Comparable interface or create a Comparator in order to add them to the Collection?

A. ArrayList and ArrayDeque

B. HashMap and HashSet

C. HashMap and TreeMap

D. TreeMap and TreeSet

45. What is a common reason for a stream pipeline not to run?

A. The source doesn’t generate any items.

B. There are no intermediate operations.

C. The terminal operation is missing.

D. None of the above

46. We want this code to print the titles of each book twice. Why doesn’t it?

LinkedList<String> list = new LinkedList<>();

list.add("Grapes of Wrath");

list.add("1984");

list.forEach(System.out::println);

Iterator it = list.iterator();

while (it.hasMore())

System.out.println(it.next());

A. The generic type of Iterator is missing.

B. The hasMore() method should be changed to hasNext().

C. The iteration code needs to be moved before the forEach() since the stream is used up.

D. None of the above. The code does print each book title twice.

47. What is the result of the following?

ArrayList<Integer> list = new ArrayList<>();

list.add(56);

list.add(56);

list.add(3);

TreeSet<Integer> set = new TreeSet<>(list);

System.out.print(set.size());

System.out.print(" " );

System.out.print(set.iterator().next());

A. 2 3

B. 2 56

C. 3 3

D. 3 56

48. What best describes a reduction?

A. An intermediate operation where it filters the stream it receives

B. An intermediate operation where it mathematically divides each element in the stream

C. A terminal operation where a single value is generated by reading each element in the prior step in a stream pipeline

D. A terminal operation where one element is returned from the prior step in a stream pipeline without reading all the elements

49. What is the output of the following?

5: ArrayDeque<Integer> d = new ArrayDeque<>();

6: d.offer(18);

7: d.offer(5);

8: d.push(13);

9: System.out.println(d.poll() + " " + d.poll());

A. 13 18

B. 18 5

C. 18 13

D. None of the above

50. What is the output of the following?

class Magazine {

private String name;

public Magazine(String name) {

this.name = name;

}

public int compareTo(Magazine m) {

return name.compareTo(m.name);

}

public String toString() {

return name;

}

}

public class Newstand {

public static void main(String[] args) {

Set<Magazine> set = new TreeSet<>();

set.add(new Magazine("highlights"));

set.add(new Magazine("Newsweek"));

set.add(new Magazine("highlights"));

System.out.println(set.iterator().next());

}

}

A. highlights

B. Newsweek

C. The code does not compile.

D. The code compiles but throws an exception at runtime.

51. What is the result of the following?

6: List<String> list = new ArrayList<>();

7: list.add("Monday");

8: list.add(String::new);

9: list.add("Tuesday");

10: list.remove(0);

11: System.out.println(list.get(0));

A. An empty String

B. Monday

C. The code does not compile.

D. The code compiles but throws an exception at runtime.

52. How many lines does this code output?

List<String> list = new LinkedList<>();

list.add("Archie");

list.add("X-Men");

list.stream().forEach(s -> System.out.println(s));

list.stream().forEach(s -> System.out.println(s));

A. Two

B. Four

C. The code does not compile.

D. The code compiles but throws an exception at runtime.

53. Which line in the main() method doesn’t compile or points to a class that doesn’t compile?

1: interface Comic<C> {

2: void draw(C c);

3: }

4: class ComicClass<C> implements Comic<C> {

5: public void draw(C c) {

6: System.out.println(c);

7: }

8: }

9: class SnoopyClass implements Comic<Snoopy> {

10: public void draw(Snoopy c) {

11: System.out.println(c);

12: }

13: }

14: class SnoopyComic implements Comic<Snoopy> {

15: public void draw(C c) {

16: System.out.println(c);

17: }

18: }

19: public class Snoopy {

20: public static void main(String[] args) {

21: Comic<Snoopy> c1 = c > System.out.println(c);

22: Comic<Snoopy> c2 = new ComicClass<>();

23: Comic<Snoopy> c3 = new SnoopyClass();

24: Comic<Snoopy> c4 = new SnoopyComic();

25: }

26: }

A. Line 21

B. Line 22

C. Line 23

D. Line 24

54. Which of the following fills in the blank so that the code outputs one line but uses a poor practice?

import java.util.\*;

public class Cheater {

int count = 0;

public void sneak(Collection<String> coll) {

coll.stream().\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

}

public static void main(String[] args) {

Cheater c = new Cheater();

c.sneak(Arrays.asList("weasel"));

}

}

A. peek(System.out::println)

B. peek(System.out::println).findFirst()

C. peek(r -> System.out.println(r)).findFirst()

D. peek(r -> {count++; System.out.println(r); }).findFirst()

55. What is the result of the following?

List<Double> list = new ArrayList<>();

list.add(5.4);

list.add(1.2);

Optional<Double> opt = list.stream().sorted().findFirst();

System.out.println(opt.get() + " " + list.get(0));

A. 1.2 1.2

B. 1.2 5.4

C. 5.4 5.4

D. None of the above

56. What is the output of the following application?

package paper;

import java.io.Closeable;

public class PrintCompany {

class Printer implements Closeable { // r1

public void print() {

System.out.println("This just in!");

}

public void close() {}

}

public void printHeadlines() {

try {Printer p = new Printer()} { // r2

p.print();

}

}

public static void main(String[] headlines) {

new PrintCompany().printHeadlines(); // r3

}

}

A. This just in!

B. The code does not compile because of line r1.

C. The code does not compile because of line r2.

D. The code does not compile because of line r3.

57. Which statement about try-with-resources is not true?

A. If the try block and close() method both throw an exception, the one thrown by

the close() method is suppressed.

B. A catch block is not required.

C. If more than one resource is used, the resources are closed in the order they were

created.

D. Parentheses are used for the resource declaration section, even if more than one

resource is used.

58. How many lines of text does the following program print?

package bee;

class SpellingException extends RuntimeException {}

public class SpellChecker {

public final static void main(String... participants) {

try {

if(!"cat".equals("kat")) {

new SpellingException();

}

} catch (SpellingException | NullPointerException e) {

System.out.println("Spelling problem!");

} catch (Exception e) {

System.out.println("Unknown Problem!");

} finally {

System.out.println("Done!");

}

}

}

A. One

B. Two

C. Three

D. The code does not compile.

59. Which exception classes, when inserted into the blank in the Problems class, allow the

code to compile?

package more;

class MissingMoneyException extends Exception {}

class MissingFoodException extends Exception {}

public class Problems {

public void doIHaveAProblem() throws MissingMoneyException,

MissingFoodException {

System.out.println("No problems");

}

public static void main(String[] lots) throws \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_{

try {

final Problems p = new Problems();

p.doIHaveAProblem();

} catch (Exception e) {

throw e;

}

}

}

I. Exception

II. MissingMoneyException

III. MissingMoneyException, MissingFoodException

A. I only

B. III only

C. I and III

D. I, II, and II

60. Which statement about Closeable and AutoCloseable is true?

A. AutoCloseable extends Closeable.

B. The close() method in a class that implements AutoCloseable cannot throw an

IOException.

C. The close() method in a class that implements Closeable cannot throw an

Exception.

D. There is no difference; one was added for backward compatibility.