**Introduction:**

**6.** What is the result of the following program?

1: public class MathFunctions {

2: public static void addToInt(int x, int amountToAdd) {

3: x = x + amountToAdd;

4: }

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

6: int a = 15;

7: int b = 10;

8: MathFunctions.addToInt(a, b);

9: System.out.println(a); } }

**A.** 10

**B.** 15

**C.** 25

**D.** Compiler error on line 3.

**E.** Compiler error on line 8.

**F.** None of the above.

**7.** What is the result of the following code?

int[] array = {6,9,8};

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

list.add(array[0]);

list.add(array[2]);

list.set(1, array[1]);

list.remove(0);

System.out.println(list);

**A.** [8]

**B.** [9]

**C.** Something like [Ljava.lang.String;@160bc7c0

**D.** An exception is thrown.

**E.** The code does not compile.

**8.** What is the output of the following code?

1: public class Deer {

2: public Deer() { System.out.print("Deer"); }

3: public Deer(int age) { System.out.print("DeerAge"); }

4: private boolean hasHorns() { return false; }

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

6: Deer deer = new Reindeer(5);

7: System.out.println(","+deer.hasHorns());

8: }

9: }

10: class Reindeer extends Deer {

11: public Reindeer(int age) { System.out.print("Reindeer"); }

12: public boolean hasHorns() { return true; }

13: }

**A.** DeerReindeer,false

**B.** DeerReindeer,true

**C.** ReindeerDeer,false

**D.** ReindeerDeer,true

**E.** DeerAgeReindeer,false

**F.** DeerAgeReindeer,true

**G.** The code will not compile because of line 7.

**H.** The code will not compile because of line 12.

**9.** Which of the following statements are true? (Choose all that apply)

**A.** Checked exceptions are intended to be thrown by the JVM (and not the programmer).

**B.** Checked exceptions are required to be caught or declared.

**C.** Errors are intended to be thrown by the JVM (and not the programmer).

**D.** Errors are required to be caught or declared.

**E.** Runtime exceptions are intended to be thrown by the JVM (and not the programmer).

**F.** Runtime exceptions are required to be caught or declared.

**10.** Which are true of the following code? (Choose all that apply)

1: import java.util.\*;

2: public class Grasshopper {

3: public Grasshopper(String n) {

4: name = n;

5: }

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

7: Grasshopper one = new Grasshopper("g1");

8: Grasshopper two = new Grasshopper("g2");

9: one = two;

10: two = null;

11: one = null;

12: }

13: private String name; }

**A.** Immediately after line 9, no grasshopper objects are eligible for garbage collection.

**B.** Immediately after line 10, no grasshopper objects are eligible for garbage collection.

**C.** Immediately after line 9, only one grasshopper object is eligible for garbage collection.

**D.** Immediately after line 10, only one grasshopper object is eligible for garbage collection.

**E.** Immediately after line 11, only one grasshopper object is eligible for garbage collection.

**F.** The code compiles.

**G.** The code does not compile.

**Java Building Blocks**

**6.** Given the following classes, what is the maximum number of imports that can be removed

and have the code still compile?

package aquarium; public class Water { }

package aquarium;

import java.lang.\*;

import java.lang.System;

import aquarium.Water;

import aquarium.\*;

public class Tank {

public void print(Water water) {

System.out.println(water); } }

**A.** 0

**B.** 1

**C.** 2

**D.** 3

**E.** 4

**F.** Does not compile.

**7.** Given the following classes, which of the following snippets can be inserted in place of

INSERT IMPORTS HERE and have the code compile? (Choose all that apply)

package aquarium;

public class Water {

boolean salty = false;

}

package aquarium.jellies;

public class Water {

boolean salty = true;

}

package employee;

INSERT IMPORTS HERE

public class WaterFiller {

Water water;

}

**A.** import aquarium.\*;

**B.** import aquarium.Water;

import aquarium.jellies.\*;

**C.** import aquarium.\*;

import aquarium.jellies.Water;

**D.** import aquarium.\*;

import aquarium.jellies.\*;

**E.** import aquarium.Water;

import aquarium.jellies.Water;

**F.** None of these imports can make the code compile.

**8.** Given the following class, which of the following calls print out Blue Jay? (Choose all that

apply)

public class BirdDisplay {

public static void main(String[] name) {

System.out.println(name[1]);

} }

**A.** java BirdDisplay Sparrow Blue Jay

**B.** java BirdDisplay Sparrow "Blue Jay"

**C.** java BirdDisplay Blue Jay Sparrow

**D.** java BirdDisplay "Blue Jay" Sparrow

**E.** java BirdDisplay.class Sparrow "Blue Jay"

**F.** java BirdDisplay.class "Blue Jay" Sparrow

**G.** Does not compile.

**9.** Which of the following legally fill in the blank so you can run the main() method from the

command line? (Choose all that apply)

public static void main( )

**A.** String[] \_names

**B.** String[] 123

**C.** String abc[]

**D.** String \_Names[]

**E.** String... $n

**F.** String names

**G.** None of the above.

**10.** Which of the following are legal entry point methods that can be run from the command

line? (Choose all that apply)

**A.** private static void main(String[] args)

**B.** public static final main(String[] args)

**C.** public void main(String[] args)

**D.** public static void test(String[] args)

**E.** public static void main(String[] args)

**F.** public static main(String[] args)

**G.** None of the above.

**Operators and Statements**

What is the output of the following code snippet?

3: int x = 4;

4: long y = x \* 4 - x++;

5: if(y<10) System.out.println("Too Low");

6: else System.out.println("Just right");

7: else System.out.println("Too High");

**A.** Too Low

**B.** Just Right

**C.** Too High

**D.** Compiles but throws a NullPointerException.

**E.** The code will not compile because of line 6.

**F.** The code will not compile because of line 7.

**7.** What is the output of the following code?

1: public class TernaryTester {

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

3: int x = 5;

4: System.out.println(x > 2 ? x < 4 ? 10 : 8 : 7);

5: }}

**A.** 5

**B.** 4

**C.** 10

**D.** 8

**E.** 7

**F.** The code will not compile because of line 4.

**8.** What is the output of the following code snippet?

3: boolean x = true, z = true;

4: int y = 20;

5: x = (y != 10) ^ (z=false);

6: System.out.println(x+", "+y+", "+z);

**A.** true, 10, true

**B.** true, 20, false

**C.** false, 20, true

**D.** false, 20, false

**E.** false, 20, true

**F.** The code will not compile because of line 5.

**9.** How many times will the following code print "Hello World"?

3: for(int i=0; i<10 ; ) {

4: i = i++;

5: System.out.println("Hello World");

6: }

**A.** 9

**B.** 10

**C.** 11

**D.** The code will not compile because of line 3.

**E.** The code will not compile because of line 5.

**F.** The code contains an infinite loop and does not terminate.

**10.** What is the output of the following code?

3: byte a = 40, b = 50;

4: byte sum = (byte) a + b;

5: System.out.println(sum);

**A.** 40

**B.** 50

**C.** 90

**D.** The code will not compile because of line 4.

**E.** An undefined value.

**Core Java APIs**

**6.** What is the result of the following code?

public class Lion {

public void roar(String roar1, StringBuilder roar2) {

roar1.concat("!!!");

roar2.append("!!!");

}

public static void main(String[] args) {

String roar1 = "roar";

StringBuilder roar2 = new StringBuilder("roar");

new Lion().roar(roar1, roar2);

System.out.println(roar1 + " " + roar2);

} }

**A.** roar roar

**B.** roar roar!!!

**C.** roar!!! roar

**D.** roar!!! roar!!!

**E.** An exception is thrown.

**F.** The code does not compile.

**7.** Which are the results of the following code? (Choose all that apply)

String letters = "abcdef";

System.out.println(letters.length());

System.out.println(letters.charAt(3));

System.out.println(letters.charAt(6));

**A.** 5

**B.** 6

**C.** c

**D.** d

**E.** An exception is thrown.

**F.** The code does not compile.

**8.** Which are the results of the following code? (Choose all that apply)

String numbers = "012345678";

System.out.println(numbers.substring(1, 3));

System.out.println(numbers.substring(7, 7));

System.out.println(numbers.substring(7));

**A.** 12

**B.** 123

**C.** 7

**D.** 78

**E.** A blank line.

**F.** An exception is thrown.

**G.** The code does not compile.

**9.** What is the result of the following code?

3: String s = "purr";

4: s.toUpperCase();

5: s.trim();

6: s.substring(1, 3);

7: s += " two";

8: System.out.println(s.length());

**A.** 2

**B.** 4

**C.** 8

**D.** 10

**E.** An exception is thrown.

**F.** The code does not compile.

**10.** What is the result of the following code? (Choose all that apply)

13: String a = "";

14: a += 2;

15: a += 'c';

16: a += false;

17: if ( a == "2cfalse") System.out.println("==");

18: if ( a.equals("2cfalse")) System.out.println("equals");

**A.** Compile error on line 14.

**B.** Compile error on line 15.

**C.** Compile error on line 16.

**D.** Compile error on another line.

**E.** ==

**F.** equals

**G.** An exception is thrown.

**Methods and Encapsulation**

6. Which of the following are true? (Choose all that apply)

A. Package private access is more lenient than protected access.

B. A public class that has private fields and package private methods is not visible to classes outside the package.

C. You can use access modifiers so only some of the classes in a package see a particular package private class.

D. You can use access modifiers to allow read access to all methods, but not any instance variables.

E. You can use access modifiers to restrict read access to all classes that begin with the word Test.

7. Given the following my.school.ClassRoom and my.city.School class definitions, which line numbers in main() generate a compiler error? (Choose all that apply)

1: package my.school;

2: public class Classroom { }

3: private int roomNumber;

4: protected String teacherName;

5: static int globalKey = 54321;

6: public int floor = 3;

7: Classroom(int r, String t) {

8: roomNumber = r;

9: teacherName = t; } }

8. Which of the following are true? (Choose all that apply)

A. Encapsulation uses package private instance variables.

B. Encapsulation uses private instance variables.

C. Encapsulation allows setters.

D. Immutability uses package private instance variables.

E. Immutability uses private instance variables.

F. Immutability allows setters.

9. Which are methods using JavaBeans naming conventions for accessors and mutators? (Choose all that apply)

A. public boolean getCanSwim() { return canSwim;}

B. public boolean canSwim() { return numberWings;}

C. public int getNumWings() { return numberWings;}

D. public int numWings() { return numberWings;}

E. public void setCanSwim(boolean b) { canSwim = b;}

10. What is the output of the following code?

1: package rope;

2: public class Rope {

3: public static int LENGTH = 5;

4: static {

5: LENGTH = 10;

6: }

**Class Design**

6. Choose the correct statement about the following code:

1: interface HasExoskeleton {

2: abstract int getNumberOfSections();

3: }

4: abstract class Insect implements HasExoskeleton {

5: abstract int getNumberOfLegs();

6: }

7: public class Beetle extends Insect {

8: int getNumberOfLegs() { return 6; }

9: }

A. It compiles and runs without issue.

B. The code will not compile because of line 2.

C. The code will not compile because of line 4.

D. The code will not compile because of line 7.

E. It compiles but throws an exception at runtime.

7. Which of the following statements about polymorphism are true? (Choose all that apply)

A. A reference to an object may be cast to a subclass of the object without an explicit cast.

B. If a method takes a superclass of three objects, then any of those classes may be passed as a parameter to the method. C. A method that takes a parameter with type java.lang.Object will take any reference.

D. All cast exceptions can be detected at compile-time.

E. By defining a public instance method in the superclass, you guarantee that the specific method will be called in the parent class at runtime.

8. Choose the correct statement about the following code:

1: public interface Herbivore {

2: int amount = 10;

3: public static void eatGrass();

4: public int chew() {

5: return 13;

6: }

7: }

A. It compiles and runs without issue.

B. The code will not compile because of line 2.

C. The code will not compile because of line 3.

D. The code will not compile because of line 4.

E. The code will not compile because of lines 2 and 3.

F. The code will not compile because of lines 3 and 4.

9. Choose the correct statement about the following code:

1: public interface CanFly {

2: void fly();

3: }

4: interface HasWings {

5: public abstract Object getWindSpan();

6: }

7: abstract class Falcon implements CanFly, HasWings {

8: }

A. It compiles without issue.

B. The code will not compile because of line 2.

C. The code will not compile because of line 4.

D. The code will not compile because of line 5.

E. The code will not compile because of lines 2 and 5.

F. The code will not compile because the class Falcon doesn’t implement the interface methods.

10. Which statements are true for both abstract classes and interfaces? (Choose all that apply)

A. All methods within them are assumed to be abstract.

B. Both can contain public static final variables.

C. Both can be extended using the extend keyword.

D. Both can contain default methods.

E. Both can contain static methods.

F. Neither can be instantiated directly.

G. Both inherit java.lang.Object.

**Exceptions**

6. What will happen if you add the statement System.out.println(5 / 0); to a working main() method?

A. It will not compile.

B. It will not run.

C. It will run and throw an ArithmeticException.

D. It will run and throw an IllegalArgumentException.

E. None of the above.

7. What is printed besides the stack trace caused by the NullPointerException from line 16?

1: public class DoSomething {

2: public void go() {

3: System.out.print("A");

4: try {

5: stop();

6: } catch (ArithmeticException e) {

7: System.out.print("B");

8: } finally {

9: System.out.print("C");

10: }

11: System.out.print("D");

12: }

13: public void stop() {

14: System.out.print("E");

15: Object x = null;

16: x.toString();

17: System.out.print("F");

18: }

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

20: new DoSomething().go();

21: }

22: }

A. AE

B. AEBCD

C. AEC

D. AECD

E. No output appears other than the stack trace.

8. What is the output of the following snippet, assuming a and b are both 0?

3: try {

4:

return a / b;

5: } catch (RuntimeException e) {

6: return -1;

7: } catch (ArithmeticException e) {

8: return 0;

9: } finally {

10: System.out.print("done");

11: }

A. -1

B. 0

C. done-1

D. done0

E. The code does not compile.

F. An uncaught exception is thrown.

9. What is the output of the following program?

1: public class Laptop {

2: public void start() {

3: try {

4: System.out.print("Starting up ");

5: throw new Exception();

6: } catch (Exception e) {

7: System.out.print("Problem ");

8: System. exit (0);

9: } finally {

10: System.out.print("Shutting down ");

11: }

12: }

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

14: new Laptop().start();

15: } }

A. Starting up

B. Starting up Problem

C. Starting up Problem Shutting down.

D. Starting up Shutting down E. The code does not compile. F. An uncaught exception is thrown.

10. What is the output of the following program?

1: public class Dog {

2: public String name;

3: public void parseName() {

4: System.out.print("1");

5: try {

6: System.out.print("2");

7: int x = Integer. parseInt (name);

8: System.out.print("3");

9: } catch (NumberFormatException e) {

10: System.out.print("4");

11: }

12: }

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

14: Dog leroy = new Dog();

15: leroy.name = "Leroy";

16: leroy.parseName();

17: System.out.print("5");

18: } }

A. 12 B. 1234 C. 1235 D. 124 E. 1245 F. The code does not compile. G. An uncaught exception is thrown.