**Introduction:**

**1.** What is the result of the following class? (Choose all that apply)

1: public class \_C {

2: private static int $;

3: public static void main(String[] main) {

4: String a\_b;

5: System.out.print($);

6: System.out.print(a\_b);

7: } }

**A.** Compiler error on line 1.

**B.** Compiler error on line 2.

**C.** Compiler error on line 4.

**D.** Compiler error on line 5.

**E.** Compiler error on line 6.

**F.** 0null

**G.** nullnull

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

String s1 = "Java";

String s2 = "Java";

StringBuilder sb1 = new StringBuilder();

sb1.append("Ja").append("va");

System.out.println(s1 == s2);

System.out.println(s1.equals(s2));

System.out.println(sb1.toString() == s1);

System.out.println(sb1.toString().equals(s1));

**A.** true is printed out exactly once.

**B.** true is printed out exactly twice.

**C.** true is printed out exactly three times.

**D.** true is printed out exactly four times.

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

**3.** What is the output of the following code? (Choose all that apply)

1: interface HasTail { int getTailLength(); }

2: abstract class Puma implements HasTail {

3: protected int getTailLength() {return 4;}

4: }

5: public class Cougar extends Puma {

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

7: Puma puma = new Puma();

8: System.out.println(puma.getTailLength());

9: }

10:

11: public int getTailLength(int length) {return 2;}

12: }

**A.** 2

**B.** 4

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

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

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

**F.** The code will not compile because of line 11.

**G.** The output cannot be determined from the code provided.

**4.** What is the output of the following program?

1: public class FeedingSchedule {

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

3: boolean keepGoing = true;

4: int count = 0;

5: int x = 3;

6: while(count++ < 3) {

7: int y = (1 + 2 \* count) % 3;

8: switch(y) {

9: default:

10: case 0: x -= 1; break;

11: case 1: x += 5;

12: }

13: }

14: System.out.println(x);

15: } }

**A.** 4

**B.** 5

**C.** 6

**D.** 7

**E.** 13

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

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

13: System.out.print("a");

14: try {

15: System.out.print("b");

16: throw new IllegalArgumentException();

17: } catch (RuntimeException e) {

18: System.out.print("c");

19: } finally {

20: System.out.print("d");

21: }

22: System.out.print("e");

**A.** abe

**B.** abce

**C.** abde

**D.** abcde

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

**F.** An uncaught exception is thrown.

**Java Building Blocks**

**1.** Which of the following are valid Java identifiers? (Choose all that apply)

**A.** A$B

**B.** \_helloWorld

**C.** true

**D.** java.lang

**E.** Public

**F.** 1980\_s

**2.** What is the output of the following program?

1: public class WaterBottle {

2: private String brand;

3: private boolean empty;

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

5: WaterBottle wb = new WaterBottle();

6: System.out.print("Empty = " + wb.empty);

7: System.out.print(", Brand = " + wb.brand);

8: } }

**A.** Line 6 generates a compiler error.

**B.** Line 7 generates a compiler error.

**C.** There is no output.

**D.** Empty = false, Brand = null

**E.** Empty = false, Brand =

**F.** Empty = null, Brand = null

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

4: short numPets = 5;

5: int numGrains = 5.6;

6: String name = "Scruffy";

7: numPets.length();

8: numGrains.length();

9: name.length();

**A.** Line 4 generates a compiler error.

**B.** Line 5 generates a compiler error.

**C.** Line 6 generates a compiler error.

**D.** Line 7 generates a compiler error.

**E.** Line 8 generates a compiler error.

**F.** Line 9 generates a compiler error.

**G.** The code compiles as is.

**4.** Given the following class, which of the following is true? (Choose all that apply)

1: public class Snake {

2:

3: public void shed(boolean time) {

4:

5: if (time) {

6:

7: }

8: System.out.println(result);

9:

10: }

11: }

**A.** If String result = "done"; is inserted on line 2, the code will compile.

**B.** If String result = "done"; is inserted on line 4, the code will compile.

**C.** If String result = "done"; is inserted on line 6, the code will compile.

**D.** If String result = "done"; is inserted on line 9, the code will compile.

**E.** None of the above changes will make the code compile.

**5.** Given the following classes, which of the following can independently replace INSERT

IMPORTS HERE to make the code compile? (Choose all that apply)

package aquarium;

public class Tank { }

package aquarium.jellies;

public class Jelly { }

package visitor;

INSERT IMPORTS HERE

public class AquariumVisitor {

public void admire(Jelly jelly) { } }

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

**B.** import aquarium.\*.Jelly;

**C.** import aquarium.jellies.Jelly;

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

**E.** import aquarium.jellies.Jelly.\*;

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

**Operators and Statements**

**1.** Which of the following Java operators can be used with boolean variables? (Choose all that

apply)

**A.** ==

**B.** +

**C.** --

**D.** !

**E.** %

**F.** <=

**2.** What data type (or types) will allow the following code snippet to compile? (Choose all that

apply)

byte x = 5;

byte y = 10;

\_\_\_\_\_ z = x + y;

**A.** int

**B.** long

**C.** boolean

**D.** double

**E.** short

**F.** byte

**3.** What is the output of the following application?

1: public class CompareValues {

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

3: int x = 0;

4: while(x++ < 10) {}

5: String message = x > 10 ? "Greater than" : false;

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

7: }

8: }

**A.** Greater than,10

**B.** false,10

**C.** Greater than,11

**D.** false,11

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

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

**4.** What change would allow the following code snippet to compile? (Choose all that apply)

3: long x = 10;

4: int y = 2 \* x;

**A.** No change; it compiles as is.

**B.** Cast x on line 4 to int.

**C.** Change the data type of x on line 3 to short.

**D.** Cast 2 \* x on line 4 to int.

**E.** Change the data type of y on line 4 to short.

**F.** Change the data type of y on line 4 to long.

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

3: java.util.List<Integer> list = new java.util.ArrayList<Integer>();

4: list.add(10);

5: list.add(14);

6: for(int x : list) {

7: System.out.print(x + ", ");

8: break;

9: }

**A.** 10, 14,

**B.** 10, 14

**C.** 10,

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

**E.** The code will not compile because of line 8.

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

**Core Java APIs**

**1.** What is output by the following code? (Choose all that apply)

1: public class Fish {

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

3: int numFish = 4;

4: String fishType = "tuna";

5: String anotherFish = numFish + 1;

6: System.out.println(anotherFish + " " + fishType);

7: System.out.println(numFish + " " + 1);

8: } }

**A.** 4 1

**B.** 41

**C.** 5

**D.** 5 tuna

**E.** 5tuna

**F.** 51tuna

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

**2.** Which of the following are output by this code? (Choose all that apply)

3: String s = "Hello";

4: String t = new String(s);

5: if ("Hello".equals(s)) System.out.println("one");

6: if (t == s) System.out.println("two");

7: if (t.equals(s)) System.out.println("three");

8: if ("Hello" == s) System.out.println("four");

9: if ("Hello" == t) System.out.println("five");

**A.** one

**B.** two

**C.** three

**D.** four

**E.** five

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

**3.** Which are true statements? (Choose all that apply)

**A.** An immutable object can be modified.

**B.** An immutable object cannot be modified.

**C.** An immutable object can be garbage collected.

**D.** An immutable object cannot be garbage collected.

**E.** String is immutable.

**F.** StringBuffer is immutable.

**G.** StringBuilder is immutable.

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

7: StringBuilder sb = new StringBuilder();

8: sb.append("aaa").insert(1, "bb").insert(4, "ccc");

9: System.out.println(sb);

**A.** abbaaccc

**B.** abbaccca

**C.** bbaaaccc

**D.** bbaaccca

**E.** An exception is thrown.

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

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

2: String s1 = "java";

3: StringBuilder s2 = new StringBuilder("java");

4: if (s1 == s2)

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

6: if (s1.equals(s2))

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

**A.** 1

**B.** 2

**C.** 12

**D.** No output is printed.

**E.** An exception is thrown.

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

**Methods and** Encapsulation

1. Which of the following can fill in the blank in this code to make it compile? (Choose all that apply) public class Ant { \_\_\_\_\_ void method() { } }

A. default

B. final

C. private

D. Public E.

String F.

2. Which of the following compile? (Choose all that apply)

A. final static void method4() { }

B. public final int void method() { }

C. private void int method() { }

D. static final void method3() { }

E. void final method() {}

F. void public method() { }

3. Which of the following methods compile? (Choose all that apply)

A. public void methodA() { return;}

B. public void methodB() { return null;}

C. public void methodD() {}

D. public int methodD() { return 9;}

E. public int methodE() { return 9.0;}

F. public int methodF() { return;}

G. public int methodG() { return null;}

4. Which of the following compile? (Choose all that apply)

A. public void moreA(int... nums) {}

B. public void moreB(String values, int... nums) {}

C. public void moreC(int... nums, String values) {}

D. public void moreD(String... values, int... nums) {}

E. public void moreE(String[] values, ...int nums) {}

F. public void moreF(String... values, int[] nums) {}

G. public void moreG(String[] values, int[] nums) {}

5. Given the following method, which of the method calls return 2? (Choose all that apply) public int howMany(boolean b, boolean... b2) { return b2.length; }

A. howMany();

B. howMany(true);

C. howMany(true, true);

D. howMany(true, true, true);

E. howMany(true, {true});

F. howMany(true, {true, true});

G. howMany(true, new boolean[2]);

**Class Design**

1. What modifiers are implicitly applied to all interface methods? (Choose all that apply)

A. protected

B. public

C. static

D. void

E. abstract

F. default

2. What is the output of the following code?

1: class Mammal {

2: public Mammal(int age) {

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

4: }

5: }

6: public class Platypus extends Mammal {

7: public Platypus() {

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

9: }

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

11: new Mammal(5);

12: }

13: }

A. Platypus

B. Mammal

C. PlatypusMammal

D. MammalPlatypus

E. The code will not compile because of line 8.

F. The code will not compile because of line 11.

3. Which of the following statements can be inserted in the blank line so that the code will compile successfully? (Choose all that apply)

public interface CanHop {}

public class Frog implements CanHop {

public static void main(String[] args) {

frog = new TurtleFrog();

} }

public class BrazilianHornedFrog extends Frog {}

public class TurtleFrog extends Frog {}

A. Frog

B. TurtleFrog

C. BrazilianHornedFrog

D. CanHop

E. Object

F. Long

4. Which statement(s) are correct about the following code? (Choose all that apply)

public class Rodent {

protected static Integer chew() throws Exception

{ System.out.println("Rodent is chewing");

return 1;

} }

public class Beaver extends Rodent {

public Number chew() throws RuntimeException {

System.out.println("Beaver is chewing on wood"); return 2;

} }

A. It will compile without issue.

B. It fails to compile because the type of the exception the method throws is a subclass of the type of exception the parent method throws.

C. It fails to compile because the return types are not covariant.

D. It fails to compile because the method is protected in the parent class and public in the subclass.

E. It fails to compile because of a static modifier mismatch between the two methods.

5. Which of the following may only be hidden and not overridden? (Choose all that apply)

A. private instance methods

B. protected instance methods

C. public instance methods

D. static methods

E. public variables

F. private variables

**Exceptions**

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

A. Runtime exceptions are the same thing as checked exceptions.

B. Runtime exceptions are the same thing as unchecked exceptions.

C. You can declare only checked exceptions.

D. You can declare only unchecked exceptions.

E. You can handle only Exception subclasses.

2. Which of the following pairs fill in the blanks to make this code compile? (Choose all that apply)

7: public void ohNo() \_\_\_\_\_ Exception {

8: \_\_\_\_\_\_\_\_\_\_\_\_\_ Exception();

9: }

A. On line 7, fill in throw

B. On line 7, fill in throws

C. On line 8, fill in throw

D. On line 8, fill in throw new

E. On line 8, fill in throws

F. On line 8, fill in throws new

3. When are you required to use a finally block in a regular try statement (not a try-withresources)?

A. Never.

B. When the program code doesn’t terminate on its own.

C. When there are no catch blocks in a try statement.

D. When there is exactly one catch block in a try statement.

E. When there are two or more catch blocks in a try statement.

4. Which exception will the following throw?

Object obj = new Integer(3); String str = (String) obj; System.out.println(str);

A. ArrayIndexOutOfBoundsException

B. ClassCastException

C. IllegalArgumentException

D. NumberFormatException

E. None of the above.

5. Which of the following exceptions are thrown by the JVM? (Choose all that apply)

A. ArrayIndexOutOfBoundsException

B. ExceptionInInitializerError

C. java.io.IOException

D. NullPointerException

E. NumberFormatException