OCJP Java SE 11 Programmer Questions

1. Given:

```
void myLambda () {
int i = 25;
Supplier<Integer> foo= () -> i;
i++;
System.out.println (foo.get());
}
```

Which is true?

- ◯ The code throws an exception at runtime.
- ◯ The code prints 25.
- [✓ ***The code does not compile.***]
- ◯ The code compiles but does not print any result.

2. Given:

```
public class Tester {
public static void main(String[] args) {
StringBuilder sb = new StringBuilder (5) ;
sb.append("HOWDY");
sb.insert (0, ' ');
sb.replace (3, 5, "LL");
sb.insert (6, "COW");
sb.delete (2, 7);
System.out.println (sb.length ());
}
```

What is the result?

- ◯ An exception is thrown at runtime.

- [✓ ***5***]
- 4
- 3

3. Given:

```
package b;
public class Person {
protected Person () {
//line 1
}
}
```

And

```
package a;
import b. Person;
public class Main {
//line 2
public static void main(String[] args) {
Person person = new Person (); //line 3
}
}
```

Which two allow a.Main to allocate a new Person? (Choose two.)

- In Line 2, change the access modifier to protectedprotected class Main {
- [✓ ***In Line 1, change the access modifier to publicpublic Person() {***]
- to create a new Main objectPerson person = new Main();
- In Line 1, change the access modifier to privateprivate Person() {
- [✓ ***In Line 2, add extends Person to the Main classpublic class Main extends Person {and change Line***]
- In Line 1, remove the access modifierPerson() {

4. Given:

```
public class Price {
private final double value; public Price (String value) {
this (Double.parseDouble (value));
}
public Price (double value) { this.value = value;
}
public Price () {}
public double getValue() { return value; }
public static void main(String[] args) {
Price pl = new Price ("1.99");
Price p2 = new Price (2.99);
Price p3= new Price ();
System.out.println (pl.getValue()+", "+p2.getValue ()+", "+p3.getValue());
}
```

What is the result?

- ○ 1.99,2.99
- [✓ ● ***The compilation fails***]
- ○ 1.99,2.99,0
- ○ 1.99,2.99,0.0

5. Given:

```
package test.t1;
public class A {
public int x = 42;
protected A() {}
}
```

And

```
// line 1
package test.t2;
```

```
import test.t1.*;
public class B extends A {
int x = 17;
public B() {
super();
}
// line 2
// line 3
}
```

And

```
package test;
import test.t1.*;
import test.t2.*;
public class Tester {
public static void main(String[] args) {
A obj = new B ();
// line 4
System.out.println (obj.x); // line 5
}
```

What is the result?

- ○ The compilation fails due to an error in line 1.
- ○ The compilation fails due to an error in line 4
- ○ The compilation fails due to an error in line 5
- ○ 17
- ○ The compilation fails due to an error in line 3
- [✓ ***42***]

6. Given:

```
public interface InterfaceOne {
void printOne();
```

}

Which three classes successfully override printOne()? (Choose three.)

A.

```
public abstract class TestClass implements InterfaceOne {
public abstract void printone ();
}
```

B.

```
public class Testclass implements InterfaceOne {
private void printone () {
System.out.println("one");
}
}
```

C.

```
public class TestClass implements InterfaceOne {
public void printone () {
System.out.println("one");
}
```

D.

```
public abstract class TestClass implements InterfaceOne {
public void printone () {
System.out.println("one");
}
```

- ☐ Option B
- ☑ ***Option C***
- ☑ ***. Option A***

- [✓] ***Option D***

7. Given:

```
import java.util. function. BiFunction;
public class Pair<T> {
final BiFunction<T, T, Boolean> validator;
T left = null;
T right = null;
private Pair() {
validator=null;
}
Pair (BiFunction<T, T, Boolean> v, T x, T y) {
validator = v;
set (x, y);
}
void set (T x, T y) {
if (!validator.apply(x, y)) throw new IllegalArgumentException ();
setLeft (x);
setRight (y);
}
void setLeft (T x) {
left = x;
}
void setRight (T y) {
right = y;
}
final boolean isValid () {
return validator.apply(left, right);
```

It is required that if p instanceof Pair then p.isValid() returns true.

Which is the smallest set of visibility changes to insure this requirement is met?

- [✓] ***left and right must be private.***
- left, right, setLeft, and setRight must be private.
- setLeft and setRight must be protected.
- isValid must be public.

8. Given:

```
public class Test {
private String[] strings;
}
```

Which two constructors will compile and set the class field strings? (Choose two.)

A.

```
public Test (List<String> strings) {
this.strings = strings;
}
```

B.

```
public Test (String... strings) {
strings = strings;
}
```

C.

```
public Test (String... strings) {
this.strings = strings;
}
```

D.

```
public Test (String strings) {
strings = strings;
}
```

E.

```
public Test (String[] strings) {
this.strings = strings;
}
```

- ☐ Option B
- ☐ Option A
- ☑ ***<u>Option C</u>***
- ☐ Option D
- ☑ ***<u>Option E</u>***

9. Given the code fragment:

```
int[] secA = { 2, 4, 6, 8, 10 };
int[] secB = { 2, 4, 8, 6, 10 };
int res1 = Arrays.mismatch (secA, secB);
int res2 = Arrays.compare (secA, secB);
System.out.print (res1 + ": " + res2);
```

What is the result?

- ○ 0.043055556
- ● ***<u>2 : -1</u>***
- ○ 0.085416667
- ○ 0.125

10. Given:

```
private int x;
public class Tester {
private static int y;
```

```
public static void main(String[] args) {
Tester t1 = new Tester ();
t1.x = 2;
Tester.y = 3;
Tester t2 = new Tester ();
t2.x = 4;
t2.y = 5;
System.out.println (t1.x+", "+t1.y);
System.out.println (t2.x+", "+Tester.y);
System.out.println (t2.x+", "+t1.y);
}
```

What is the result?

- [✓ ***2,54,54,5***]
- 2,34,54,3
- 2,34,54,5
- 2,34,34,5

11. Given:

```
1. interface Pastry {
2. void getIngredients ();
3. }
4. abstract class Cookie implements Pastry {
   }
5.
6. class ChocolateCookie implements Cookie {
7. public void getIngredients    () {
   }
8. }
9. class Coconut ChocolateCookie extends ChocolateCookie {
10. void     getIngredients (int x) {
     }
11. }
```

Which is true?

- ○ The compilation fails due to an error in line 4.
- ○ The compilation fails due to an error in line 2
- ○ The compilation fails due to an error in line 10
- ○ The compilation fails due to an error in line 7
- ○ The compilation succeeds.
- [✓ ***<u>The compilation fails due to an error in line 6</u>***]

12. Given:

```
public class Hello {

public static void main(String[] args) {

System.out.println (args [0] +args [1] +args [2]);

}

}
```

executed using command:

java Hello “Hello World” Hello World What is the output?

- ○ Hello WorldHello World
- ○ Hello WorldHelloWorld
- ○ An exception is thrown at runtime
- ○ HelloHello WorldHelloWorld
- [✓ ***<u>Hello World Hello World</u>***]

13. Given:

```
public class A {

private boolean checkValue (int val) {

return true;

}

}

And

public class B extends A {

public int modifyVal (int val) {

if (checkValue (val)) {

return val;

} else {

return 0;

}

public static void Main(String[] args) {

B b = new B();

System.out.println(b.modifyval (10) ) ;

}
```

What is the result?

- [✓ ***It fails to compile.***]
- 10

- ○ A java.lang.IllegalArgumentException is thrown
- ○ nothing

14. Given:

```
public class Over {

public void analyze (Object [] 0) {

System.out.println("I am an object array");

}

public void analyze (long [] 1) {

System.out.println("I am an array");

}

public void analyze (Object o) {

System.out.println("I am an object");

}

public static void main(String[] args) {

int[] nums = new int [10];

new Over ().analyze (nums); // line 1

}

}
```

What is the output?

- ○ I am an object array

- ○ The compilation fails due to an error in line 1.
- [✓ ***I am an object***]
- ○ I am an array

15. Given:

```
public class Foo {

public <T> Collection<T> foo (Collection<T> arg) { …… }

}
```

And

```
public class Bar extends Foo { …… }
```

Which two statements are true if the method is added to Bar? (Choose two.)

- ☐ public Collection foo(Collection arg) { ... } overloads Foo.foo.
- [✓ ***public Iterable foo(Collection arg) { ... } overrides Foo.foo***]
- ☐ public Collection foo(Collection arg) { ... } overrides Foo.foo.
- ☐ public Collection foo(Stream arg) { ... } overloads Foo.foo.
- ☐ public Collection bar(Collection arg) { ... } overloads Foo.foo
- [✓ ***public List foo(Collection arg) { ... } overrides Foo.foo.***]

16. Given:

```
int x = 0;

while (x < 10) {

System.out.print (x++);
}
```

Which “for” loop produces the same output?

A.

```
int b = 0;
for ( b < 10; ) {
System.out.print (++b);
}
```

B.

```
for (a; a< 10; a++) {
System.out.print (a);
}
```

C.

```
for (int d = 0; d < 10; ) {
System.out.print (d);
++d;
}
```

D.

```
for (int c = 0; ; c++) {
break;
if (c = 10) {
System.out.print (c);
}

}
```

- [✓ ***Option C***]
- Option B
- Option A
- Option D

17. Given:

```
public class Test {

public static void main(String[] args) {

int x;

int y = 5;

if (y > 2) {

x = ++y;

y = x + 7;

} else {

y++;

}

System.out.print (x + + y);

}
```

What is the result?

- ○ 0 5
- ○ 6 13
- ○ 5 12
- [✓ ***<u>compilation error</u>***]

18. Given:

```
public interface API { //line 1

public void checkValue (Object value)

throws IllegalArgumentException; //line 2

public boolean isValueANumber (Object val) {

if(val instanceof Number) {

return true;

}

else {

try {

Double.parseDouble (val.toString());

return true;

}catch (Number FormatException ex) {

return false;

}

}

}
}
```

Which two changes need to be made to make this class compile? (Choose two.)

- ☐ Change Line 1 to an abstract class:public abstract class API {
- ☐ Change Line 1 to extend java.lang.AutoCloseable:public interface API extends AutoCloseable {
- ☐ Change Line 2 access modifier to protected:protected void checkValue(Object value)throws IllegalArgumentException;
- ☑ ***Change Line 1 to a class:public class API {***
- ☑ ***Change Line 2 to an abstract method:public abstract void checkValue(Object value)throws IllegalArgumentException;***

19. Given:

```
import java.util.ArrayList;

import java.util.Arrays;

public class NewMain {

public static void main(String[] args) {

String[] fruitNames = { "apple", "orange",

"grape", "lemon", "apricot", "watermelon" };

var fruits = new ArrayList<> (Arrays.asList (fruitNames));

fruits.sort ((var a, var b) -> -a.compareTo (b));

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

}

}
```

What is the result?

- appleapricotgrapelemonorangewatermelon
- appleorangegrapelemonapricotwatermelon
- nothing
- [✓ ***watermelonorangelemongrapeapricotapple***]

20. Given:

```
class Myclass {

public static void main(String [] args) {

System.out.println (arg [1] + "--" + arg[3] + "--" + arg[0]);

}

}
```

executed using this command: java Myclass My Car is red What is the output of this class

- Myclass--Car--re
- My--is--java
- [✓ ***My--Car--is***]
- java--Myclass--My
- Car--red—My

21. Given:

```
public class DNASynth {

int aCount;
int tCount;

int cCount;

int gCount;

DNASynth (int a, int tCount, int c, int g) {

// line 1

}

int setCCount (int c) {

return c;

}

void setGCount (int gCount) {

this.gCount = gCount;

}

}
```

Which two lines of code when inserted in line 1 correctly modifies instance variables? (Choose two.)

- ☐ setGCount(g);
- ☐ cCount = setCCount(c);
- [✓] ***<u>aCount = a;</u>***
- [✓] ***<u>tCount = tCount;</u>***
- ☐ setCCount(c) = cCount;

22. Given:

```
public static void main(String[] args) {

char letter = 'b';

int i = 0;

switch (letter) {

case 'a':

i++;

break;

public class Tester {

case 'b':

i++;

case 'c' I 'd': // line 1

i++;

case 'e':

i++;

break;

case 'f':

i++;

break;

default:

System.out.print (letter);

}
System.out.println (i);

}

}
```

What is the result?

- [✓ ***3***]
- B2
- 1
- 2
- b3
- b1

23. Given the code fragment:

```
char [] [] arrays = {{'a', 'd'}, {'b', 'e'}, {'c', 'f'}};

for (char[] xx: arrays) {

for (char yy xx) {

System.out.print (yy);

}

System.out.print(" ");

}
```

What is the result?

- [✓ ***ad be cf***]
- The compilation fails.
- ab cd ef

- ○ abc def
- ○ An ArrayIndexOutOfBoundsException is thrown at runtime

24. Given:

```
class Employee {
String office;
}
```

And the code fragment:

```
5. public class HRApp {
6. var employee = new ArrayList<Employee> ();
7. public var display() {
8. var employee = new Employee ();
9. var offices = new ArrayList<> ();
10. offices.add("Chicago");
11. offices.add("Bangalore");
12. for (var office offices) {
13. System.out.print ("Employee Location"+ office);
14.}
15. }
16. }
```

Which two lines cause compilation errors? (Choose two.)

- ☐ line 12
- ☑ ***line 6***
- ☐ line 8
- ☐ line 9
- ☑ ***line 7***

25. Given:

```
public interface Euler Interface {

double getEulerValue ();

}

public class EulerLambda {

public static void main(String[] args) {

Euler Interface myEuler Interface;

myEuler Interface = () -> "2.71828";

System.out.println("Value of Euler = " + myEuler Interface.getEulerValue ()
);

}

}
```

What is the result?

- ○ Value of Euler = 2.71828
- ○ Value of Euler = “2.71828”
- ○ It throws a runtime exception
- ● ***The code does not compile.***

26. Given:

```
public class Test {

public static void main(String[] args) {

AnotherClass ac = new AnotherClass();

SomeClass sc = new AnotherClass();

ac = sc;

sc.methodA () ;

ac.methodA ();

}

}

class SomeClass {

public void methodA ()

System.out.println("SomeClass#methodA ()") ;

}

class AnotherClass extends SomeClass {

public void methodA () {

System.out.println("AnotherClass#methodA () ");

}

}
```

What is the result?

- A ClassCastException is thrown at runtime.
- SomeClass#methodA()SomeClass#methodA()
- [✓ ***The compilation fails.***]
- SomeClass#methodA()AnotherClass#methodA()
- AnotherClass#methodA()AnotherClass#methodA()
- AnotherClass#methodA()SomeClass#methodA()

27. Given:

```
String[] [] arr = {

{"Red", "White"},

{"Black"},

{"Blue", "Yellow", "Green", "Violet"}

};

for (int row = 0; row < arr.length; row++) {

int column = 0;

for (; column < arr[row].length; column++) {

System.out.println("[" + row + "," + column + "] = " + arr[row] [column]);

}

}
```

What is the result?

- [0,0] = Red[0,1] = White[1,0] = Black[1,1] = Blue[2,0] = Yellow[2,1] = Green[3,0]   = Violet

- [✓ ***[0,0] = Red[0,1] = White[1,0] = Black[2,0] = Blue[2,1] = Yellow[2,2] = Green[2,3] = Violet***]
- [0,0] = Red[1,0] = Black[2,0] = Blue
- java.lang.ArrayIndexOutOfBoundsException thrown

28. Given:

```
public class Test{

private int num = 1;

private int div = 0;

public void divide () {

try {

num = num / div;

System.out.print("Exception");

}

catch (ArithmeticException ae) { num = 100; }

catch (Exception e) { num = 200; }

finally { num = 300; }

System.out.print (num);

}

public static void main(String args[])

{

Test test = new Test();
test.divide ();

}

}
```

What is the output?

- ○ Exception
- ○ 200
- ○ ***<u>300</u>***
- ○ 100

29. Given:

```
class Mycar {

}
```

and

```
javac C:\workspace4\Mycar.java
```

What is the expected result of javac?

- ○ javac fails to compile the class and prints the error message, C:\ workspace4\Mycar.java:1:error: package java does not exist
- [✓ ● ***<u>javac compiles Mycar.java without errors or warnings.</u>***]
- ○ javac fails to compile the class and prints the error message, Error : Could not find or load main class Mycar.class
- ○ javac fails to compile the class and prints the error message, C:\w orkspace4\Mycar.java:1:error: expected import java.lang

30. Given:

```
class ConSuper {

protected ConSuper () {

this (2);

System.out.print("1");

}

protected ConSuper (int a) {

System.out.print (a);

}

}
```

and

```
public class ConSub extends ConSuper {

ConSub () {

this (4);

System.out.print("3");

}

ConSub (int a) {

System.out.print (a);

}

public static void main (String[] args) {

new ConSub (4);
}

}
```

What is the result?

- ○ 2143
- ○ 2134
- [✓ ***<u>214</u>***]
- ○ 234

31. Given:

```
public class Tester {

public static void main(String[] args) {

String s= "this is it";

int x = s.indexOf("is");

s.substring (x+3) ;

x = s.indexOf("is");

System.out.println (s+" "+x);

}

}
```

What is the result?

- ○ this is it 3
- ○ is it 0
- ○ An IndexOutOfBoundsException is thrown at runtime.

- [✓ ***this is it 2***]
- is it 1

32. Given:

```
public class Foo {

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

for (var x : args) {

System.out.println (x) ;

}

}

}
```

What is the type of the local variable x?

- Character
- String[ ]
- char
- [✓ ***String***]

33. Given:

```
public interface A {

abstract void x();

}
```

and

```
public abstract class B /* position 1 */ {

/* position 2 */

public abstract void z();

public void x() { }

}

and

public class C extends B implements A {

/* position 3 */

}
```

Which code, when inserted at one or more marked positions, would allow classes B and C to compile?

- ○ @Override // position 3void x () {} // position 3@Override // position 3public void z() { } // position 3
- [✓ ***<u>@Override // position 2public void z() { } // position 3</u>***]
- ○ public void z() { } // position 3
- ○ implements A // position 1@Override // position 2

34. Given the code fragment:

```
String s1 = new String ("ORACLE");

String s2 "ORACLE";

String s3 = s1.intern();

System.out.print ((s1==s2) + ");
System.out.print((s2==s3) +

System.out.println (s1==s3);

");
```

What is the result?

- ○ false false true
- ○ true false false
- ○ false true true
- [✓ ***<u>false true false</u>***]

35. Given:

```
StringBuilder S = new StringBuilder ("ABCD");
```

Which would cause s to be AQCD?

- ○ s.replace(s.indexOf(“A”), s.indexOf(“C”), “Q”)
- ○ s.replace(s.indexOf(“B”), s.indexOf(“B”), “Q”)
- [✓ ***<u>s.replace(s.indexOf(“B”), s.indexOf(“C”), “Q”)</u>***]
- ○ s.replace(s.indexOf(“A”), s.indexOf(“B”), “Q”);

36. Given:

```
import java.io.*;

public class Tester {

public static void main(String[] args) {

try {
```

```
doA ();

doB ();

} catch (IOException e) {

System.out.print("c");

return;

} finally{

System.out.print("d");

}

System.out.print("f");

}

private static void doA() {

System.out.print ("a");

if (false) {

throw new IndexOutOfBoundsException ();

}

}

private static void doB () throws FileNotFoundException {
System.out.print("b");

if (true) {

throw new FileNotFoundException();

}
```

```
}

}
```

What is the result?

- ad
- abd
- [✓ ***abcd***]
- The compilation fails.
- abd

37. Given:

```
1. {
2. Iterator iter = List.of (1,2,3).iterator ();
3. while (iter.hasNext()) {
4. foo (iter.next());
5. }
6. Iterator iter2 = List.of (1,2,3).iterator ();
7. while (iter.hasNext()) {
8. bar (iter2.next());
9. }
10. }
11. for (Iterator iter = List.of (1,2,3). iterator (); iter.hasNext(); ) {
12. foo (iter.next());
13. }

14. for (Iterator iter2 = List.of (1,2,3). iterator (); iter.hasNext (); ) {

15. bar (iter2.next());

16. }
```

Which loop incurs a compile time error?

- [✓ ***the loop starting line 14***]
- the loop starting line 11
- the loop starting line 3
- the loop starting line 7

38. Given:

```
var i = 10;

var j = 5;

i += (j* 5+j) / 1 - 2;

System.out.println(i);
```

What is the result?

- 3
- 5
- 25
- 23
- [✓ ***11***]

39. Given:

```
import java.time. LocalDate;

import static java.time. DayOfWeek. *;

public class Main {

public static void main(String[] args) {

var today = LocalDate.now().with (TUESDAY).getDayOfWeek ();

switch (today) {

case SUNDAY:

case SATURDAY:

System.out.println("Weekend");

break;

case MONDAY: FRIDAY:

System.out.println("Working");

default:

System.out.println("Unknown") ;

}

}

}
```

What is the result?

- ○ TuesdayUnknown
- [✓ ***Unknown***]
- ○ WorkingUnknown
- ○ Working

- ○ The compilation fails.
- ○ Tuesday

40. Given the code fragment:

```
int x = 0;

do {

x++;

if (x == 1) {

continue;

}

System.out.println (x);

} while (x < 1);
```

What is the result?

- ○ It prints 1 in the infinite loop
- [✓ ***The program prints nothing***]
- ○ 1
- ○ 1
- ○ 0

41. Given:

```
public class DNASynth {

int acount;
int tCount;

int cCount;

int gCount;

void setACount (int cCount) {

cCount = ccount;

}

void setTCount() {

this.tCount = tCount;

}

int setCCount() {

return cCount;

}

int setGCount (int g) {

gCount = g;

}

return gCount;

void setAllCounts (int x) {

aCount = tCount = this.cCount = setGCount (x);

}

}
```

Which two methods modify field values? (Choose two.)

- ☐ setCCount
- ☑ ***setAllCounts***
- ☐ setACount
- ☐ setTCount
- ☑ ***setGCount***

42. Given:

    public interface Example Interface { }

Which two statements are valid to be written in this interface? (Choose two.)

- ☐ public int x;
- ☐ public void methodF(){System.out.println(“F”);}
- ☐ final void methodG(){System.out.println(“G”);}
- ☑ ***. public abstract void methodB();***
- ☐ final void methodE();
- ☑ ***public String methodD();***

43. Given:

```
public class Tester {

public static void main(String[] args) {

byte x = 7, y = 6;

// line 1

System.out.println(z);

}
```

}

Which expression when added at line 1 will produce the output of 1.17?

- ◯ ***float z = Math.round((float)x/y*100)/(float)100;***
- ◯ float z = Math.round((float)x/y,2);
- ◯ float z = Math.round((int)(x/y),2);
- ◯ float z = (float)(Math.round((float)x/y*100)/100);

44. Given:

```
interface MyInterfacel {

public int method () throws Exception;

private void pMethod () { /* an implementation of pMethod */ }

}

interface MyInterface2 {

public static void sMethod () { /* an implementation of sMethod */ }

public boolean equals();

}

interface MyInterface3 {

public void method ();

public void method (String str);

}

interface MyInterface4 {
```

```
public void dMethod () { /* an implementation of dMethod */ }

public void method ();

}

interface MyInterface5 {

public static void sMethod ();

public void method (String str);

}
```

Which two interfaces can be used in lambda expressions? (Choose two.)

- [✓] ***MyInterface5***
- [ ] MyInterface3
- [✓] ***MyInterface2***
- [ ] MyInterface4
- [ ] MyInterface1

45. Given:

```
public class Main {

public static void checkConfiguration (String filename) {

File file = = new File (filename);

if (!file.exists()) {

throw new Error ("Fatal Error: Configuration File, + filename + ", is
missing.");

}
}

public static void main(String[] args) {

checkConfiguration ("App.config");

System.out.println("Configuration is OK");

}

}
```

If file “App.config” is not found, what is the result?

- ○ Configuration is OK
- ○ Exception in thread "main" java.lang.Error:Fatal Error: Configuration File, App.config, is missing.
- ○ nothing
- [✓ ***The compilation fails.***]

46. Given:

```
public class Person {

private String name;

public Person (String name) {

this.name = name;

}

public String toString() {

return name;

}
}

and

public class Tester {

public static void main(String[] args) {

Person p = null;

checkPerson (p);

System.out.println (p);

System.out.println (p);

p = new Person ("Mary");

checkPerson (p);

}

public static Person checkPerson (Person p) {

if (p == null) {

p = new Person ("Joe");

}else{

P = null;

}

return p;

}

}
```

What is the result?

- [✓ ***nullMary***]
- JoeMarry
- Joenull
- nullnull

47. Given:

```
public class Foo {

public void foo (Collection arg) {

System.out.println("Bonjour le monde!");

}

}
```

and

```
public class Bar extends Foo {

public void foo (Collection arg) {

System.out.println("Hello world!");

}

public void foo (List arg) {

System.out.println("Olá Mundo!");

}

}
```

and

```
Foo f1 = new Foo ().
Foo £2= new Bar ();

Bar bl = new Bar ();

Collection<String> c = new ArrayList<> ();
```

Which three are true? (Choose three.)

- ☐ b1.foo(c) prints Hello world!
- ☐ f1.foo(c) prints Olá Mundo!
- ☐ f2.foo(c) prints Olá Mundo!
- [✓ ***f2.foo(c) prints Bonjour le monde!***]
- [✓ ***b1.foo(c) prints Olá Mundo!***]
- ☐ f1.foo(c) prints Hello world

48. Given:

```
public class Foo {

private void print () {

System.out.println("Bonjour le monde ! ");

}

public void foo () {

print ();

}

}

public class Bar extends Foo {

private void print () {
```

```
System.out.println("Hello world!");

}

public void bar () {

print ();

}

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

Bar b = new Bar ();

b. foo ();

b.bar();

}

}
```

What is the output?

- ○ Hello world!Bonjour le monde!
- [✓ ***<u>Bonjour le monde!Hello world!</u>***]
- ○ Bonjour le monde!Bonjour le monde!
- ○ Hello world!Hello world!

49. Analyze the code:

```
public class Test {

static String prefix = "Global: ";

private String name = "namescope";
public static String getName() {

return new Test() .name;

}

public static void main(String[] args) {

Test t = new Test ();

System.out.println (/* Insert code here */);

}

}
```

Which two options can you insert inside println method to produce Global:namescope? (Choose two.)

- prefix+Test.name
- Test.prefix+Test.name
- Test.getName+prefix
- [✓ ***<u>Test.prefix+Test.getName()</u>***]
- prefix+name
- [✓ ***<u>new Test().prefix+new Test().name</u>***]

50. Given:

```
class Super {

static String greeting () { return "Good Night"; }

String name () { return "Harry"; }

}
```

and

```
class Sub extends Super {

static String greeting () { return "Good Morning"; }

String name () { return "Potter"; }

}

and

class Test {

public static void main(String[] args) {

Super s = new Sub ();

System.out.println (s.greeting() + 11 + s.name () );

}

}
```

What is the result?

- ○ ***<u>Good Night, Potter</u>***
- ○ Good Morning, Harry
- ○ Good Morning, Potter
- ○ Good Night, Harry

51. Given:

```
public class Main {

public static void main(String[] args) {

for (int i = 0; i < args.length; i++) {
```

```
System.out.println(i + "). " + args[i]);

switch (args[i]) {

case "one":

continue;

case "two":

i--;

continue;

default:

break;

}

}

}

}
```

executed with this command:

java Main one two three

What is the result?

- A java.lang.NullPointerException is thrown.
- 0). one1). two2). three
- [✓ ***It creates an infinite loop printing:0). one1). two1). two...***]
- The compilation fails
- 0). One

52. Given:

/code/a/Test.java containing:

```
package a;

import b.Best;

public class Test {

public static void main(String[] args) {

Best b= new Best ();

}

}
```

and

/code/b/Best.java

containing:

```
package b;

public class Best { }
```

Which is the valid way to generate bytecode for all classes?

- ○ java /code/a/Test.java
- ○ java /code/a/Test.java /code/b/Best.java
- ○ javac –d /code /code/a/Test

- ○ java –cp /code a.Test
- ○ javac –d /code /code/a/Test.java
- [✓ ***javac –d /code /code/a/Test.java /code/b/Best.java***]

53. Given the declaration:

```
@interface Resource {
String name ();
int priority () default 0;
}
```

Examine this code fragment:

```
/* Loc1 */ class ProcessOrders { … }
```

Which two annotations may be applied at Loc1 in the code fragment? (Choose two.)

- ☐ @Resource(name="Customer1")
- [✓ ***@Resource(priority=0)***]
- [✓ ***@Resource(priority=100)***]
- ☐ @Resource
- ☐ @Resource(name="Customer1", priority=100)

54. Given:

```
import java.util.*;
public class Main {
static Map<String, String> map = new HashMap<> ();
static List<String> keys =
        new ArrayList<> (List.of ("A", "B", "C", "D"));
static String[] values =
        {"one", "two", "three", "four" };
static {
for (var i = 0; i < keys.size () ; i++) {
map.put (keys.get (i), values [i]);
}
}
public static void main(String[] args) {
keys.clear();
values = new String [0];
System.out.println("Map: " + map.size() +
              " Keys: " + keys.size() +
              " Values: " + values.length);
}
}
```

What is the result?

- ○ The compilation fails.
- ○ Map: 4 Keys: 4 Values: 4
- ○ Map: 0 Keys: 0 Values: 0
- [✓ ***Map: 4 Keys: 0 Values: 0***]
- ○ Map: 0 Keys: 4 Values: 4

55. Given:

```
import java.io.FileNot FoundException;
import java.io. IOException;
public class Tester {
public static void main(String[] args) {
doA ();
try {
} //line 1
}
private static void doA () throws IOException, IndexOutOfBoundsException
{
if (false) {
```

```
throw new FileNotFoundException ();
} else {
throw new IndexOutOfBounds Exception ();
}
}
}
```

What must be added in line 1 to compile this class?

- [✓ ***<u>catch(IOException e) { }</u>***]
- ○ catch(FileNotFoundException | IndexOutOfBoundsException e) { }
- ○ catch(FileNotFoundException e) { }catch(IndexOutOfBoundsException e) { }
- ○ catch(IndexOutOfBoundsException e) { }catch(FileNotFoundException e) { }
- ○ catch(FileNotFoundException | IOException e) { }

56. Given:

```
List<String> list = ... ;
list.forEach(x -> System.out.println(x); });
```

What is the type of x?

- ○ List
- ○ char
- [✓ ***<u>String</u>***]
- ○ List

57. Given this enum declaration:

```
1. enum Alphabet {
```

```
2. A, B, C
3.
4. }
```

Examine this code:

```
System.out.println(Alphabet.getFirstLetter());
```

What code should be written at line 3 to make this code print A?

- ○ String getFirstLetter() { return A.toString(); }
- ○ static String getFirstLetter() { return Alphabet.values()[1].toString(); }
- ○ final String getFirstLetter() { return A.toString(); }
- [✓ ***static String getFirstLetter() { return A.toString(); }***]

58. Given:

```
public class Main {
public static void main(String[] args) {
int i = 1;
for (Strings : args) {
System.out.println ((i++) + ") " + s);
}
}
}
```

executed with this command:

```
java Main one two three
```

What is the output of this class?

- ○ The compilation fails.

- ○ A java.lang.ArrayIndexOutOfBoundsException is thrown
- ○ thing
- ○) one
- [✓ ***1) one2) two3) three***]

59. Given the formula to calculate a monthly mortgage payment

    M = P. [r(1+r) / (1+r)-1]

    and these declarations:

```
double m;                    //monthly payment
double r = 0.05/12;       //monthly interest rate
int p = 100_000;          //principal
int n 180;                  //number of payments
```

How can you code the formula?

- ○ m = p * r * Math.pow(1 + r, n) / Math.pow(1 + r, n) - 1;
- [✓ ***m = p * (r * Math.pow(1 + r, n) / (Math.pow(1 + r, n) - 1));***]
- ○ m = p * (r * Math.pow(1 + r, n) / Math.pow(1 + r, n) - 1);
- ○ m = p * ((r * Math.pow(1 + r, n) / (Math.pow(1 + r, n)) - 1));

60. Given:

```
public class Person {
private String name = "Joe Bloggs";
public Person (String name) {
this.name = name;
}
public String toString() {
return name;
}
}

And

public class Tester {
public static void main(String[] args) {
Person pl = new Person (); // line 1
System.out.println (p1);
}
}
```

What is the result?

- ○ Joe Bloggs
- ○ p1
- [✓ ***<u>The compilation fails due to an error in line 1</u>***]

61. Given the code fragment:

```
Path currentFile = Paths.get("/scratch/exam/temp.txt");

Path outputFile = Paths get("/scratch/exam/new.txt");

Path directory = Paths.get("/scratch/");

Files.copy(currentFile, outputFile);

Files.copy(outputFile, directory);

Files.delete (outputFile);
```

The /scratch/exam/temp.txt file exists. The /scratch/exam/new.txt and /scrat ch/new.txt files do not exist.

What is the result?

- ○ The program throws a FileaAlreadyExistsException.
- ○ A copy of /scratch/exam/new.txt exists in the /scratch directory and /scratch/exam/new.txt is deleted.
- [✓ ***<u>The program throws a NoSuchFileException.</u>***]
- ○ /scratch/exam/new.txt and /scratch/new.txt are deleted.

62. Given:

```
package a;
public abstract class Animal {
protected abstract void walk ();
}
package b;
public abstract class Human extends Animal {
// line 1
}
```

Which two lines inserted in line 1 will allow this code to compile? (Choose two.)

- ☐ void walk(){}
- [✓ ***<u>protected void walk(){}</u>***]
- [✓ ***<u>public abstract void walk();</u>***]
- ☐ private void walk(){}
- ☐ abstract void walk();

63. Given:

```
public class Test {
private int sum;
public int compute() {
```

```
int x = 0;
while (x 3) {
sum += x++;
}
return sum;
}
public static void main(String[] args) {
Test t = new Test ();
int sum t. compute();
sum = t.compute();
t.compute();
System.out.println (sum);
}
}
```

What is the result?

- [✓ ***6***]
- 9
- 3
- An exception is thrown at runtime.

64. Examine this excerpt from the declaration of the java.se module:

```
module java.se {
…
requires transitive java.sql;
…
}
```

What does the transitive modifier mean?

- [✓ ***<u>Only a module that requires the java.se module is permitted to require the java.sql module.</u>***]
- Any module that requires the java.se module does not need to require the java.sql module.
- Any module that requires the java.sql module does not need to require the java.se module.
- Any module that attempts to require the java.se module actual ly requires the java.sql module instead.

65. Given:

```
public class Person {
private String name;
public void setName (String name) {
String title = "Dr. ";
name = title+name;
}
public String toString() {
return name;
}
}
```

And

```
public class Test {
public static void main(String args[]) {
Person p= new Person ();
p.setName ("Who");
System.out.println (p);
}
}
```

What is the result?

- An exception is thrown at runtime.

- ○ Dr. Who
- ○ Dr. Null
- ○ ***null***

66. Given this requirement:

Module vehicle depends on module part and makes its com.vehicl e package available for all other modules.

Which module-info.java declaration meets the requirement?

A
```
module vehicle {
requires part;
exports com.vehicle;
}
```

B

```
module vehicle {
requires part;
uses com.vehicle;
}
```

C

```
module vehicle{
requires part;
exports com.vehicle to part;
}
```

D

```
module vehicle {
requires com.vehicle;
exports part;
}
```

- ○ Option C
- ○ Option B
- [✓ ***<u>Option A</u>***]
- ○ Option D

67. Given: