13. Given:

```
public class StrBldr {
  static StringBuilder sb1 = new StringBuilder("yo ");
  StringBuilder sb2 = new StringBuilder("hi ");

  public static void main(String[] args) {
    sb1 = sb1.append(new StrBldr().foo(new StringBuilder("hey")));
    System.out.println(sb1);
  }

  StringBuilder foo(StringBuilder s) {
    System.out.print(s + " oh " + sb2);
    return new StringBuilder("ey");
  }
}
```

What is the result?

- oh hi hey
- A compile time error occurs.
- hey oh hi yo ey
- hey oh hi
- hey oh hi ey
- yo ey

14. Given:

```
class MyPersistenceData {
  String str;
  private void methodA() {
    System.out.println("methodA");
  }
}
```

You want to implement the `java.io.Serializable` interface to the `MyPersistenceData` class.

Which method should be overridden?

- The `readExternal` and `writeExternal` method
- Nothing
- The `writeExternal` method
- The `readExternal` method

15. Given:

```
public class Employee {
    private String name;
    private String neighborhood;
    private LocalDate birthday;
    private int salary;
}
```

and

```
List roster = new ArrayList<>(...);
Map> m = roster.stream()
// Line 1
```

Which code fragment on line 1 makes the `m` map contain the employee with the highest salary for neighborhood?

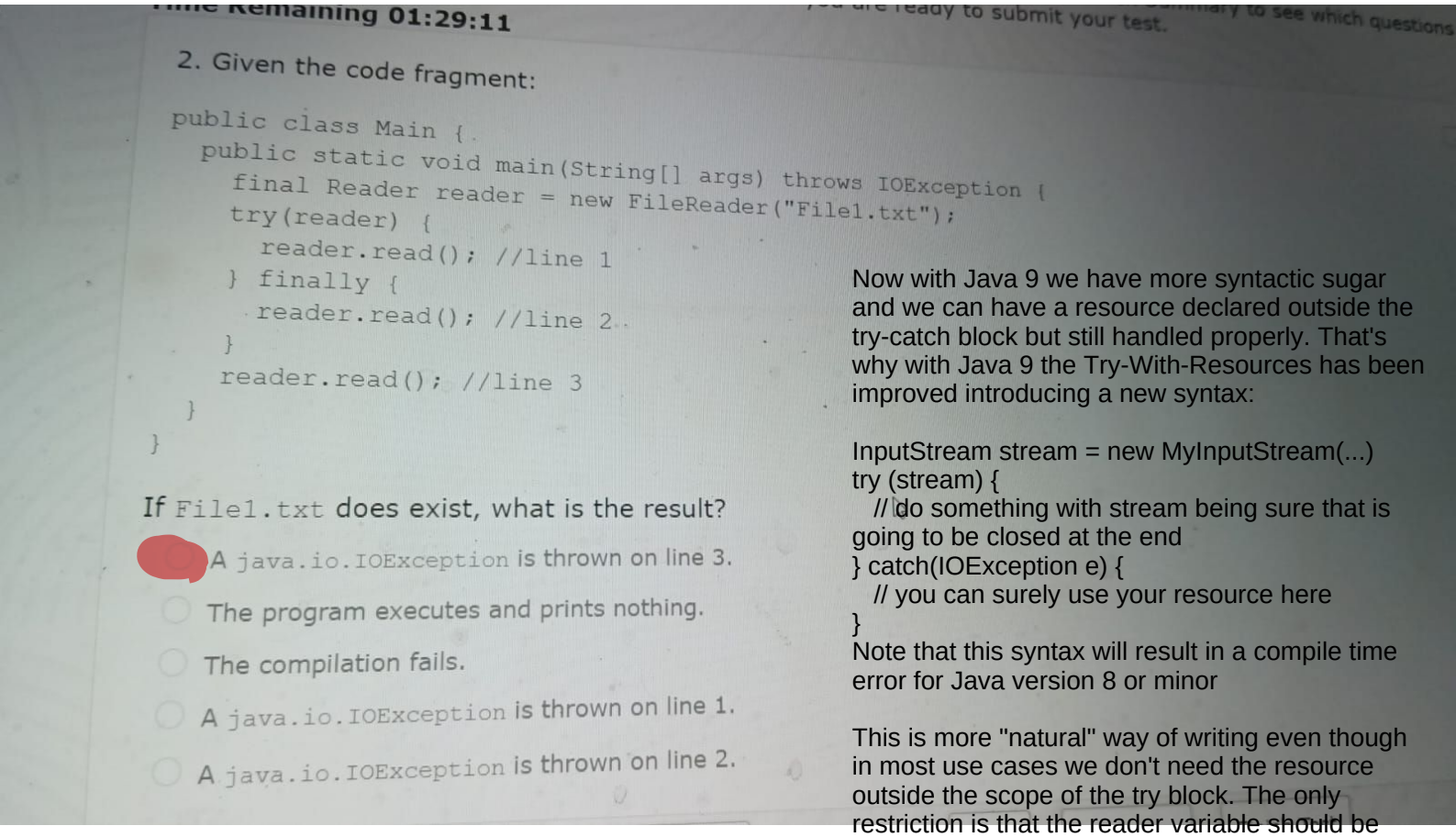
- `.collect(Collectors.groupingBy(Employee::getNeighborhood, Collectors.maxBy(Comparator.comparing(Employee::getSalary))));`
- `.collect(Collectors.groupingBy(e -> e.getNeighborhood(),`

```
public class StrBldr {
  static StringBuilder sb1 = new StringBuilder("yo ");
  static StringBuilder sb2 = new StringBuilder("hi ");

  public static void main(String[] args) {
    sb1 = sb1.append(new StrBldr().foo(new StringBuilder("hey")));
    System.out.println(sb1);
  }

  StringBuilder foo(StringBuilder s) {
    sb2 = sb2.append(s + " oh ");
    return sb2;
  }
}
```

What is the result?

- hey oh yo hi
- yo hi
- hey oh hi yo
- yo hi hey oh
- A compile time error occurs.

yo hi hey oh



Now with Java 9 we have more syntactic sugar and we can have a resource declared outside the try-catch block but still handled properly. That's why with Java 9 the Try-With-Resources has been improved introducing a new syntax:

```
InputStream stream = new MyInputStream(...)
try (stream) {
   // do something with stream being sure that is going to be closed at the end
} catch(IOException e) {
   // you can surely use your resource here
}
```

Note that this syntax will result in a compile time error for Java version 8 or minor

This is more "natural" way of writing even though in most use cases we don't need the resource outside the scope of the try block. The only restriction is that the reader variable should be effectively final or just final.

Anyway with this syntax you can surely have your resource used also in the catch and finally block

Time Remaining 01:29:36

1. Given the code fragment:

```
public class Main {
  public static void main(String[] args) {
    try {
      Path path = Paths.get("/u01/work");
      // line 1
      System.out.println(attributes.isDirectory());
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

You want to examine whether `path` is a directory.

Which code inserted on line 1 will accomplish this?

- `BasicFileAttributes attributes = Files.getAttribute(path, "isDirectory");`
- `BasicFileAttributes attributes = Files.readAttributes(path, BasicFileAttributes.class);`
- `BasicFileAttributes attributes = Files.isDirectory(path);`
- `BasicFileAttributes attributes = Files.readAttributes(path, FileAttributes.class);`

```
List roster = new ArrayList<>(...);
Map> m = roster.stream()
// Line 1
```

Which code fragment on line 1 makes the `m` map contain the employee with the highest salary for each neighborhood?

```
.collect(Collectors.groupingBy(Employee::getNeighborhood,
Collectors.maxBy(Comparator.comparing(Employee::getSalary))));
```

```
.collect(Collectors.groupingBy(e -> e.getNeighborhood(),
    Collectors.maxBy((x, y) -> y.getSalary() - x.getSalary())));
```

x - y will give the highest salary.

```
.collect(Collectors.maxBy((x, y) -> y.getSalary()
- x.getSalary(),
    Collectors.groupingBy(Employee::getNeighborhood)));
```

```
.collect(Collectors.maxBy(Employee::getSalary,
    Collectors.groupingBy(Comparator.comparing(e ->
e.getNeighborhood()))));
```

```
    }
}
System.out.println(txt1);
```

Which two statements inserted independently at line 1 enable this code to print PRRT?

- [ ] `continue b;`
- [ ] `j--;`
- [ ] `i--;`
- [ ] `continue a;`
- [ ] `break b;`
- [ ] `break a;`

Time Remaining 01:27:55

test. Click Next to go to the next test page. Click Summary to see which questions you n
Click Finish Test if you are ready to submit your test.

4. Examine:

```
Class.forName(JDBC_DRIVER_CLASS_NAME);
```

When is it necessary to execute this statement?

- It must be executed before each call to `DriverManager` to get a `Connection` using the named JDBC driver.
- It must be executed once and only before the first call to `DriverManager` to get a `Connection` using the named JDBC driver.
- It must be executed once and before accessing the named JDBC driver in any way.
- It is no longer required to execute this method.

Previous | Page 4 of 50 | Next | Summary | Finish Test

3. Given the code fragment:

```
StringBuilder txt1 = new StringBuilder("PPQRRRSTT");
int i = 0;
a:
while (i < txt1.length()) {
   char x = txt1.charAt(i);
   int j = 0;
   i++;
   b:
   while (j < txt1.length()) {
      char y = txt1.charAt(j);
      if (i != j && y == x) {
         txt1.deleteCharAt(j);
         // line 1
      }
      j++;
   }
}
System.out.println(txt1);
```

Which two statements inserted independently at line 1 enable this code to print PRRT?

- [ ] continue b;
- [ ] j--;

the destination directory.

Which code inserted on line 1 will accomplish this?

```
try {
Files.move(Paths.get(source), Paths.get(destination));
```

```
try (FileChannel in = new
FileInputStream(source).getChannel();
     FileChannel out = new FileOutputStream(destination).getChannel()) {
 in.transferTo(0, in.size(), out);
```

```
try {
Files.move(Paths.get(source), Paths.get(destination),
StandardCopyOption.REPLACE_EXISTING);
```

```
try {
Files.copy(Paths.get(source), Paths.get(destination),
StandardOpenOption.CREATE_NEW);
Files.delete(Paths.get(source));
```

6. Given the code fragment:

```
8. public class Test {
9.    private final int x = 1;
10.    static final int y;
11.    public Test() {
12.       System.out.print(x);
13.       System.out.print(y);
14.    }
15.    public static void main(String args[]) {
16.       new Test();
17.    }
18. }
```

What is the result?

- The compilation fails at line 16.
- The compilation fails at line 9.
- 10
- The compilation fails at line 13.
- 1

5. Given:

```
public class Main {
   public static void main(String[] args) {
     String source = "/u01/work/stage/message.txt";
     String destination = "/u01/work/message.txt";

     // line 1

     } catch (IOException e) {
       e.printStackTrace();
     }
   }
}
```

You want to move `source.txt` to the destination directory even if a file with the same nam
the destination directory.

Which code inserted on line 1 will accomplish this?

○ try {

Time Remaining 01:26:15

8. Which three initialization statements are correct?

- [ ] `float x = 1f;`
- [ ] `int[][][] e = {{1,1,1},{2,2,2}};`
- [ ] `String contact# = "(+2) (999) (232)";`
- [ ] `boolean false = (4 != 4);`
- [ ] `short sh = (short)'A';`
- [ ] `int x = 12_34;`
- [ ] `byte b = 10;`
  `char c = b;`

Time Remaining 01:26:31

7. Given the code fragment:

```
Integer i = 11;
```

Which two statements compile?

- [x] `Double b = Double.valueOf(i);`
- [ ] `double e = Double.parseDouble(i);`
- [ ] `Double a = i;`
- [ ] `Double c = (Double) i;`
- [x] `double d = i;`

and

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

What is the result?

- An exception is thrown at runtime.
- Green
- Mr. Blue
- Mr. Green

```
    public static void main(String args[]) {
        String s = "10";
        try {
            int x = 0;
            x = Integer.parseInt(s,2); // line 1
            System.out.println("X is "+x);
        } catch(NumberFormatException e) {
            System.out.println("Error parsing value of "+x); // line 2
        }
    }
}
```

What is the result?

- Error parsing value 0
- The compilation fails due to an error in line 1.
- X is 2.
- X is 10.
- The compilation fails due to an error in line 2.

Given:

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

and

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

```
public class Car extends Automobile {
                                        // line 2
    void wheels(int i) {                // line 3
        System.out.print(4);
    }
    public static void main(String[] args) {
        Automobile ob = new Car(); // line 4
        ob.wheels();
    }
}
```

What must you do so that the code prints 4?

- Replace the code in line 2 with `Car ob = new Car();`
- Remove the parameter from `wheels` method in line 3.
- Remove `abstract` keyword in line 1.
- Add `@Override` annotation at line 2.

11. Given:

Automobile.java

```
public abstract class Automobile { //line 1
    abstract void wheels();
}
```

Car.java

```
public class Car extends Automobile {                    // line 2
                                                         // line 3
    void wheels(int i) {
        System.out.print(4);
    }
    public static void main(String[] args) {
        Automobile ob = new Car(); // line 4
        ob.wheels();
    }
```

21. Given:

```
public interface Worker {
  public void doProcess();
}
```

and

```
public class HardWorker implements Worker {
  public void doProcess() {
    System.out.println("doing things");
  }
}
```

and

```
public class Cheater implements Worker {
  public void doProcess() { }
}
```

and

```
                    Worker> extends Thread { // Line 1
```

and

```
public class Main <T extends Worker> extends Thread { // Line 1
  private List<T> processes = new ArrayList<>();      // Line 2
  public void addProcess(HardWorker w) {              // Line 3
    processes.add(w);
  }
  public void run() {
    processes.forEach((p) -> p.doProcess());
  }
}
```

What needs to change to make these classes compile and still handle all types of interface `Worker`?

- Replace Line 3 with `public void addProcess(Worker w) {`
- Replace Line 3 with `public void addProcess(T w) {`
- Replace Line 1 with `public class Main extends Thread {`
- Replace Line 2 with `private List processes = new ArrayList<>();`

20. Given the code fragment:

```
Locale l = new Locale("en", "US");
LocalDate today = LocalDate.of(2018, 12, 17);
String mToday = today.format(DateTimeFormatter.ofLocalizedDate(FormatStyle.MEDIUM));
String sToday = today.format(DateTimeFormatter.ofLocalizedDate(FormatStyle.SHORT));
System.out.println(mToday);
System.out.println(sToday);
```

What is the result?

- December 17, 2018
  12/17/18
- Friday, December 17, 2018
  December 17, 2018
- 12/17/18
  Dec 17, 2018
- Dec 17, 2018
  12/17/18

23. Given:

```
public class ResourceTest {
  public static void main(String[] args){
    final MyResource res1 = new MyResource();
    MyResource res2 = new MyResource();
    try(res1 ; res2) {
      // do something
    } catch(Exception e) {}
  }
  static class MyResource implements AutoCloseable {
    public void close() throws Exception { }
  }
}
```

Which statement is true?

- The code fails to compile as `MyResource` must implement `Closeable`.
- The code compiles successfully.
- The code fails to compile as `res2` should be declared as `final`.
- The code fails to compile as `try-with-resource` needs a variable declaration such as `MyResource r1 = res1; MyResource r2 = res2;`.

```
}
```

and

```
public class ApiImpl extends AbstractAPI implements APIInterface {
   public void process() {
     System.out.println("Process()called 2.");
   }
   public static void main(String[] args) {
     var impl = new ApiImpl();
     impl.process();
   }
}
```

What is the result?

- The compilation fails.
- A `java.lang.NoSuchMethodException` is thrown.
- The program prints `Process()called 1.`
- The program prints `Process()called 2.`
- A `java.lang.IllegalAccessException` is thrown.

answer before submitting the test. Click Next to go to the next test page. Click Summary to see which que

Time Remaining 01:18:14

22. Which two statements are true about a class that is marked `@Deprecated`?

- [ ] There is always another class that can be used instead of the deprecated class.
- [ ] The author of the class wants to discourage people from using the class in any way.
- [ ] Using the class is guaranteed to cause errors at runtime.
- [ ] Using the class will cause the Java compiler to give a warning.
- [ ] The class cannot be extended.

```
    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?

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

16. Given:

```
public class Person {
    private String name;
    private int age;
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public int getAge() {
        return age;
    }
    public static void main(String args[]){
        var persons = Arrays.asList(new Person("Max", 18),
                                    new Person("Peter", 23),
                                    new Person("Pamela", 23),
                                    new Person("David", 12));
        int num = persons.stream()
                         .mapToInt(Person::getAge)
                         .filter(p -> p < 20)
                         .reduce(0, (a,b) -> a + b);
        System.out.println(num);
    }
}
```

What is the output?

Time Remaining 01:15:29

31. Given:

```
interface MyInterface1 {
    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();
```

Time Remaining 01:20:24

Click Finish Test if you are ready to submit your test.

17. Which code fragment does a service use to load the service provider with a `Print` interface?

- `private Print print = new com.service.Provider.PrintImpl();`
- `private java.util.ServiceLoader<Print> loader = ServiceLoader.load(Print.class);`
- `private java.util.ServiceLoader<Print> loader = new java.util.ServiceLoader<>();`
- `private Print print = com.service.Provider.getInstance();`

Time Remaining 01:19:56

test page. Click Summary to see which questions you need
test if you are ready to submit your test.

18. Which two statements are true about running code on the class path and the module path?

- [x] A non-modular JAR placed on the `-classpath` results in an unnamed module.
- [x] A modular JAR placed on the `--module-path` results in a named application module.
- [ ] A non-modular JAR placed on the `--module-path` results in a named application module.
- [ ] A modular JAR placed on the `-classpath` results in a named application module.
- [ ] A modular JAR placed on the `-classpath` results in an automatic module.

```
        int num = persons.stream()
                        .mapToInt(Person::getAge)
                        .filter(p -> p < 20)
                        .reduce(0, (a,b) -> a + b);
        System.out.println(num);
    }
}
```

What is the output?

- 30
- 41
- 46
- 35

```
20.         throw new A();
21.       }
22.     }
23.     catch(Exception e) { System.out.println(e);    }
24.     System.out.println("Continue...");
25.   }
26. }
```

You must define the `A` exception class. The program execution must be terminated if the condition at line 19 is `true` and an `A` exception is thrown at line 20.

Which code fragment at line `n1` defines `A` as per the requirement?

- `class A extends Exception {`
- `class A extends RuntimeException {`
- `class A extends Throwable {`
- `class A extends ArithmeticException {`

Time Remaining 01:19:43

...test if you are ready to submit your test.

19. Given the code fragment:

```
/* line n1 */
   A() {
     super ("The Mandatory Criteria Yet to Meet");
   }
}
```

```
15. public class TestCE {
16.   public static void main(String[] args)throws A {
17.     int a = 10, b = 13;
18.     try {
19.       if (a < b) {
20.         throw new A();
21.       }
22.     }
23.     catch(Exception e) { System.out.println(e);   }
24.     System.out.println("Continue...");
25.   }
26. }
```

...the test. Click Finish Test if you are ready to submit your test. ...the next test page. Click Summary to see which que

Time Remaining 01:17:20

25. Given the code fragment:

```
Supplier supplier = () -> "Hello World";
// line 1
```

Which statement on line 1 is calling the method of the `supplier` object correctly?

- `System.out.println(supplier.accept());`
- `System.out.println(supplier.test());`
- `System.out.println(supplier.get());`
- `System.out.println(supplier.apply())`

Time Remaining 01:17:03

26. Given:

```
String s = "Oracle";
Runnable r = () -> {
    System.out.println(s);
};
s = "Java";
Thread t = new Thread(r);
    t.start();
```

What is the result ?

- Compilation error
- Oracle
- An exception is thrown at run time.
- Java

24. Given:

```
int i = 10;
do {
  for(int j = i/2; j > 0; j--) {
    System.out.print(j + " ");
  }
  i-=2;
} while (i > 0);
```

What is the result?

- 5
- nothing
- 5 4 3 2 1
- 5 4 3 2 1 4 3 2 1 3 2 1 2 1 1

28. Given the code fragment:

```
public static void main(String... args) {
  String filename = "/u01/work" + args[0];
  // line n1

  // ...
}
```

You want to validate a path name before the read file. Before validation, all path names should be canonicalized.

Which code inserted on line `n1` will accomplish this?

-
```
File file = new File(filename);
String canonicalPath = file.getCanonicalPath();
FileInputStream fis = new FileInputStream(f);
```

-
```
File file = new File(filename).getAbsoluteFile();
FileInputStream fis = new FileInputStream(file);
```

-
```
Path file = Paths.get(filename);
String canonicalPath = file.normalize().toString();
FileInputStream fis = new FileInputStream(canonicalPath);
```

-
```
Path file = Paths.get(filename);
Path canonicalPath = file.toAbsolutePath().toString();
FileInputStream fis = new FileInputStream(canonicalPath);
```

29. Given the code fragment:

```
ExecutorService es = Executors.newCachedThreadPool();
es.execute(() -> System.out.print("Ping "));
// line 1
System.out.println(future.get()); // line 2
es.shutdown();
```

Which statement at line 1 will print `Ping Pong`?

- `Future<String> future = es.submit(() -> "Pong");`
-
```
Future<String> future = new Callable() {
    public String call() throws Exception {
      return "Pong";
    }
}.call();
```
-
```
Future<String> future = es.invokeAny(new Callable<String>(){
    public String call() throws Exception {
      return "Pong";
    }
});
```
- `Future<String> future = es.execute(() -> "Pong");`

```
public class A {
    int a = 0;
    int b = 0;
    int c = 0;
    public void foo(int i) {
        a += b * i;
        c -= b * i;
    }
    public void setB(int i) {
        b = i;
    }
}
```

Which makes `class A` thread safe?

- Make `setB` synchronized.
- Make `foo` synchronized.
- Class `A` is thread safe.
- Make `foo` and `setB` synchronized.
- Make `A` synchronized.

```
        }
        public abstract void methodC();
    }
```

and

```
public class ConcreteClass extends AbstractClass {
    public void methodC(String c) {
        System.out.println(c);
    }
}
```

Which three changes make this code compile?

- [ ] Remove `methodA()` from `AbstractClass`
- [ ] Implement `methodC()` in `ConcreteClass`
- [ ] Implement `methodB()` in `ConcreteClass`
- [ ] Add the keyword `abstract` to the `methodA()` and `methodB()` declarations in `InterfaceOne`
- [ ] Remove `methodA()` from `InterfaceOne`
- [ ] `InterfaceTwo` should no longer extend `AbstractClass`
- [ ] Implement `methodA()` in `ConcreteClass`

you are ready to submit your test. which questions

Time Remaining 01:15:56

30. Given:

```
public interface APIInterface {
   public default void process() { System.out.println ("Process() called 1."); }
}
```

and

```
public abstract class AbstractAPI {
   public abstract void process();
}
```

and

```
public class ApiImpl extends AbstractAPI implements APIInterface {
   public void process() {
      System.out.println("Process()called 2.");
   }
   public static void main(String[] args) {
      var impl = new ApiImpl();
      impl.process();
```

Time Remaining 01:07:11

47. Given:

```
package pac;
public class Hello{
  public static void main(String[] args) {
    Module module = Hello.class.getModule();
    System.out.println("Module: "+module);
    System.out.println("Name: " + module.getName());
    System.out.println("Descriptor: " + module.getDescriptor());
  }
}
```

Given the directory structure:

```
\Test
| Hello.java
```

Given the commands to execute at the Test directory prompt:

```
                    llo.java
```

```
Test>java -cp pac pac.Hello
```

Which statement is true?

- Execute `java --module-path pac pac.Hello` instead of `java -cp pac pac.Hello` and on execution the program prints:

```
Module: pac @<</font><</font>hash code>>
 Name: pac.Test
 Descriptor: null
```

- On execution of the given commands, the program prints:

```
Module: unnamed module @<</font><</font>hash code>>
 Name: null
 Descriptor: null
```

- Create an empty `module-info.java` file in the `Test` directory and on execution of the given commands, the program prints:

```
Module: unnamed module @<</font><</font>hash code>>
 Name: null
 Descriptor: module-info
```

- On execution of the given commands, the program prints:

```
Module: pac.Hello @<</font><</font>hash code>>
 Name: unnamed
 Descriptor: null
```

45. Given:

```
public class Sports {
   ....
   public double getRatings() {
    ....
    }
   ....
}
```

and

```
public class Football extends Sports {
   ....
   public double getRatings() {
    ....
    }
   ....
}
```

Which is the correct implementation of the `getRatings` method in the `Football` subclass?

- The subclass `getRatings` method implementation directly accesses the fields in the `Sports`
- ...() to call the base class method but

n(s) on this pag
submitting the test.

e Remaining 01:05:55

49. Given:

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

and the code fragment:

```
5. public class HRApp {
6.     var employee = new ArrayList();
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"+ of
14.        }
15.    }
16. }
```

Which two lines cause compilation errors?

- [ ] line 6
- [ ] line 12
- [ ] line 8
- [ ] line 7
- [ ] line 9

50. 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 p1 = new Person(); // line 1
        System.out.println(p1);
    }
}
```

What is the result?

- p1
- null
- The compilation fails due to an error in line 1.
- Joe Bloggs

48. Given:

```
import java.sql.Timestamp;
public class Test {
    public static void main(String[] args) {
        Timestamp ts = new Timestamp(1);
    }
}
```

and the commands:

```
javac Test.java
jdeps -summary Test.class
```

What is the result on execution of these commands?

- `Test.class -> java.base Test.class -> java.sql`
- On execution, the `jdeps` command displays an error.
- `Test.class -> java.sql -> java.base`
- `Test.class -> java.base Test.class -> java.sql java.sql -> java.base`

Click Finish Test if you are ready to submit your test. Click Summary to see y

Time Remaining 01:13:12

33. Which two can be considered good practices for serializing Java objects?

- [ ] Implement secure serialization by generating secure object hash or using encryption.
- [ ] Implement serialization for long-term data storage.
- [ ] Assign null value by default while serializing and deserializing a `transient` variable.
- [ ] Always override the `readObject/writeObject` methods from the `java.io.Serializable` interface.
- [ ] Ensure that the class definition used is the same as the class definition used by Java runtime at the time when the object was serialized.

34. Given:

```
class Item {
    public String name; public int count;
    public Item(String name, int count) {
        this.name = name; this.count = count;
    }
}
```

and the code fragment:

```
public class Test {
    public static void main(String[] args) {
        var items = List.of(new Item("A", 10),new Item("B", 2),
                            new Item("C", 12),new Item("D", 5),new Item("E", 6));

        // line 1
        System.out.println("There is an item for which the variable count is below
    }
}
}
```

... for which the variable count is below zero

32. Given:

```
public interface InterfaceOne {
     public void methodA();
     public void methodB();
 }
```

and

```
public interface InterfaceTwo extends AbstractClass {  }
```

and

```
public abstract class AbstractClass implements InterfaceOne {
     public String origin = "Abstract Class";
     public void methodA() {
         System.out.println("A");
     }
     public abstract void methodC();
 }
```

and

```
AbstractClass {
```

Time Remaining 01:12:01

...ish Test if you are ready to submit your test.

35. Why would you choose to use a `peek` operation instead of a `forEach` operation on a Stream?

- To remove an item from the end of the stream.
- To remove an item from the beginning of the stream.
- To process the current item and return a stream.
- To process the current item and return `void`.

36. Given:

```
char[] characters = new char[100];
try (FileReader reader = new FileReader("file_to_path");) {
  // line 1
    System.out.println(String.valueOf(characters));
  } catch(IOException e) {
    e.printStackTrace();
}
```

You want to read data through the `reader` object.

Which statement inserted on line 1 will accomplish this?

- `characters.read();`
- `reader.readLine();`
- `characters = reader.read();`
- `reader.read(characters);`

```
    static void main(String[] args) {
        var items = List.of(new Item("A", 10),new Item("B", 2),
                            new Item("C", 12),new Item("D", 5),new Item("E", 6));
        // line 1
        System.out.println("There is an item for which the variable count is below zero.");
    }
}
```

You want to examine the `items` list if it contains an item for which the variable count is below zero.

Which code fragment at line 1 will accomplish this?

- `f(items.stream().filter(i -> i.count < 0).findAny()) {`
- `if(items.stream().filter(i -> i.count< 0).findFirst()) {`
- `if(items.stream().allMatch(i -> i.count < 0)) {`
- `if(items.stream().anyMatch(i -> i.count < 0 )) {`

- abyssinian
  oxicat
  korat
  laperm
  bengal
  sphynx
- sphynx
  oxicat
  laperm
  korat
  bengal
  abyssinian
- abyssinian
  bengal
  korat
  laperm
  oxicat
  sphynx
- nothing

37. Given:

```
import java.util.ArrayList;
import java.util.Arrays;
public class NewMain {
  public static void main(String[] args) {
    String[] catNames = { "abyssinian", "oxicat",
         "korat", "laperm", "bengal", "sphynx" };
    var cats = new ArrayList<>(Arrays.asList(catNames));
    cats.sort((var a, var b) -> -a.compareTo(b));
    cats.forEach(System.out::println);
  }
}
```

What is the result?

- abyssinian
  oxicat
  korat
  laperm
  bengal

39. Given:

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

and

```
public class Tester {
 static Person p = null;
 public static void main(String[] args) {
  p = checkPerson(p);
  System.out.println(p);
  Person p1 = new Person("Joe");
  p1 = checkPerson(p);
  System.out.println(p1);
 }
```

```
    Person p1 = new Person("Joe");
    p1 = checkPerson(p);
    System.out.println(p1);
  }
  public static Person checkPerson(Person p) {
    if (p == null) {
      p = new Person("Mary");
    }
    return p;
  }
}
```

What is the result?

- Mary
  Mary
- Marry
  Joe
- null
  null
- Joe
  Joe

38. Given:

```
static void add(List l) {
    l.add(4);
    l.add(3.14f);
}

public static void main(String[] args) {
    var x = new ArrayList();
    x.add(3);
    add(x);
    for (Integer i : x) {
        System.out.print(i + " ");
          }
}
```

What is the result?

- The program prints `3 4` and throws a `ClassCastException`.
- `3 4 3.14`
- `3 4 3`
- `3 4`

Time Remaining 01:09:39

41. 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`?

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

```
public class Over {
    public void analyze(Object[] o){
      System.out.println("I am an object array");
    }
    public void analyze(long[] l){
      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?

- The compilation fails due to an error in line 1.
- I am an object
- I am an array
- I am an object array

Time Remaining 01:09:54

40. Given the code fragment:

```
public class DoClass {
   static String s;
   public static void main(String[] args) {
     switch(s) {
        case "41": s += "41";
        default:   s += " def ";
        case "42": s += "42";
     }
     System.out.println(s);
  }
}
```

What is the output?

- 41 def 42
- def 42
- An exception is thrown at runtime.
- null

```
public class Bar extends Foo {
    public void foo(Collection arg) {
        System.out.println("Hello world!");
    }
    public static void main(String... args) {
        List<String> li = new ArrayList<>();
        Bar b = new Bar();
        Foo f = b;
        b.foo(li);
        f.foo(li);
    }
}
```

What is the output?

- Bonjour le monde!
  Hello world!
- Hello world!
  Hello world!
- Hello world!
  Bonjour le monde!
- Bonjour le monde!
  Bonjour le monde!

44. 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("Hola Mundo!");
    }
}
```

15. 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 static void main(String... args) {
        List<String> li = new ArrayList<>();
        Bar b = new Bar();
        Foo f = b;
        b.foo(li);
        f.foo(li);
    }
}
```

```
    ....
    public double getRatings() {
    ....
    }
  ....
}
```

Which is the correct implementation of the `getRatings` method in the `Football` subclass?

- The subclass `getRatings` method implementation directly accesses the fields in the `Sports` superclass.
- The subclass `getRatings` method uses `public.getRatings()` to call the base class method but uses its own named fields in the implementation.
- The subclass `getRatings` method uses `new.getRatings()` to call the base class method but uses its own named fields in the implementation.
- The subclass `getRatings` method uses `super.getRatings()` to call the base class method but uses its own named fields in the implementation.

46. Given the code fragment:

```
public class City {
    public static void main(String[] args) {
        String[] towns = {"boston", "paris", "bangkok", "oman"};
        Comparator ms = (a, b) -> b.compareTo(a);
        Arrays.sort(towns, ms);
        System.out.println(Arrays.binarySearch(towns, "oman", ms));
    }
}
```

What is the result?

- -1
- -3
- 2
- 1

and

```
Foo f1 = new Foo();
Foo f2 = new Bar();
Bar b1 = new Bar();
List<String> li = new ArrayList<>();
```

Which three are correct?

- [ ] `f2.foo(li)` prints `Hola Mundo!`
- [ ] `b1.foo(li)` prints `Bonjour le monde!`
- [ ] `f2.foo(li)` prints `Hello world!`
- [ ] `b1.foo(li)` prints `Hello world!`
- [ ] `f2.foo(li)` prints `Bonjour le monde!`
- [ ] `b1.foo(li)` prints `Hola Mundo!`
- [ ] `f1.foo(li)` prints `Hello world!`
- [ ] `f1.foo(li)` prints `Bonjour le monde!`
- [ ] `f1.foo(li)` prints `Hola Mundo!`