1.If, after fetching a view from an activity under test, you want to ensure that the view actually exists in the view hierarchy. What would you do?

* Incorrect -

Do an assertNull check on the viewInstance.

* Incorrect -

Do an assertEquals check on the viewInstance's ID.

* Correct -

Do an assertNotNull check on the viewInstance.

* Your choice: incorrect -

Call some method on the viewInstance and catch NullPointerException.

* Incorrect -

I don't know yet.

2.Which of the following class from Android architecture components can be used to store and manage UI-related data in a lifecycle-conscious way?

* Incorrect -

AppModel

* Incorrect -

View

* Incorrect -

AppView

* Your choice: correct -

ViewModel

* Incorrect -

I don't know yet.

3.How would you perform an action on an item in RecyclerView that is not currently being shown on the screen?

* Incorrect -

Bring it on screen using onData.

* Incorrect -

Bring it on screen using onView.

* Your choice: incorrect -

Bring it on screen using scrollToViewWithValue.

* Correct -

Bring it on screen using scrollToPosition.

* Incorrect -

I don't know yet.

4.Where can you locate unit tests in an Android project?

* Correct -

module-name/src/test/java/

* Incorrect -

/src/test/java/

* Your choice: incorrect -

module-name/src/test/

* Incorrect -

module-name/test/java/

* Incorrect -

I don't know yet.

5.Which of the following is required for a test class to run? (Assume the class is in the correct package.)

* Incorrect -

One or multiple methods annotated with @Test and a method annotated with @Before

* Your choice: incorrect -

One or multiple methods annotated with @Test and methods annotated with @Before & @After

* Incorrect -

There are no requirements.

* Correct -

One or multiple methods annotated with @Test

* Incorrect -

I don't know yet.

6.If you are trying to access a view using onView and its ID and you get an exception AmbiguousViewMatcherException, what's the most likely cause for this?

* Incorrect -

There are multiple views with the same ID in the whole app.

* Correct -

There are multiple views with the same ID in the activity.

* Your choice: incorrect -

There are no views with the mentioned ID in the activity.

* Incorrect -

There are no views with the mentioned ID in the whole app.

* Incorrect -

I don't know yet.

7.If you are writing a test that needs location permission from the device, how would you ensure that the test has it when it runs?

* Incorrect -

By granting permissions for the test app using UI automator on Settings app.

* Incorrect -

By asking the system for the permission during test setup.

* Your choice: correct -

By defining a GrantPermissionRule which is assigned the value returned by the GrantPermissionRule.grant method.

* Incorrect -

By mocking granting of permissions to the test.

* Incorrect -

I don't know yet.

8.Which types of tests form the basis of the Test-driven Development (TDD) pyramid?

* Your choice: correct -

Unit tests

* Incorrect -

Integration tests

* Incorrect -

Unit or integration tests

* Incorrect -

UI tests

* Incorrect -

I don't know yet.

9.In which directory under the root folder should the UI automator test class be placed?

* Incorrect -

test

* Your choice: correct -

androidTest

* Incorrect -

automatorTest

* Incorrect -

uiTest

* Incorrect -

I don't know yet.

10.What kind of lifecycle events can we generate on an Activity instance created using Robolectric?

* Your choice: correct -

All lifecycle events

* Incorrect -

Only create/destroy

* Incorrect -

Only create/resume/start

* Incorrect -

Only create/resume

* Incorrect -

I don't know yet.

11.If our test class requires setup before each test, we would \_\_\_\_\_\_\_\_.

* Correct -

Create a setup method annotated with @Before

* Incorrect -

Create a setup method annotated with @BeforeClass

* Incorrect -

Create a setup method annotated with @SetUp

* Your choice: incorrect -

Create a setup method annotated with @Init

* Incorrect -

I don't know yet.

12.Which of the following annotations must we use to specify a method should run before each test?

* Incorrect -

@Setup

* Incorrect -

@Init

* Your choice: incorrect -

@BeforeTest

* Correct -

@Before

* Incorrect -

I don't know yet.

13.Which of the following is an example of using an Answer object in the best way?

* Your choice: incorrect -

Calling a related onSuccess() method on an interface.

* Incorrect -

Using method stubbing to return a desired value.

* Incorrect -

Returning an implementation of an interface.

* Correct -

Calling a related onSuccess() method on an async callback interface.

* Incorrect -

I don't know yet.

14.How would you simulate pressing the home button in a test?

* Incorrect -

UiAppDevice.pressHome()

* Incorrect -

Device.pressHome()

* Your choice: correct -

UiDevice.pressHome()

* Incorrect -

UiDevice.pressCenterKey()

* Incorrect -

I don't know yet.

15.Which of the following is required for a test class to run (assuming the class is in the correct package)?

* Incorrect -

One or multiple methods annotated with @Test and a method annotated with @Before

* Incorrect -

One or multiple methods annotated with @Test and methods annotated with @Before & @After

* Incorrect -

There are no requirements.

* Your choice: correct -

One or multiple methods annotated with @Test

* Incorrect -

I don't know yet.

16.Which of the following is an example of a problem that test suites solves?

* Correct -

Running an array of different test classes that are explicitly specified

* Your choice: incorrect -

You cannot create test suites when developing on Android

* Incorrect -

Running all test classes in the immediate package

* Incorrect -

Running all created test classes

* Incorrect -

I don't know yet.

17.Where should the instrumented tests that run on a device be added in an Android project?

* Your choice: incorrect -

module-name/androidTest/java/

* Incorrect -

module-name/test/java/

* Incorrect -

module-name/src/test/java/

* Correct -

module-name/src/androidTest/java/

* Incorrect -

I don't know yet.

18.Which of the following is a valid sequence of methods to call when performing action on an element inside a WebView?

* Incorrect -

perform(...).onWebView().withElement(...)

* Your choice: correct -

onWebView().withElement(...).perform(...)

* Incorrect -

withElement(...).onWebView().perform(...)

* Incorrect -

onWebView().perform(...)

* Incorrect -

I don't know yet.

1.What is the correct syntax to create a spy for an array list with the help of Mockito annotation?

* Incorrect -

@Mock

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

* Your choice: correct -

@Spy

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

* Incorrect -

@Spy

@Mock

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

* Incorrect -

List<String> spyList;

spyList = Spy.from(new ArrayList<>());

* Incorrect -

I don't know yet.

2.Using the AAA structure, after setting ALL the necessary preconditions and inputs as part of the Arrange phase, which of the following is the next step in a unit test?

* Your choice: correct -

Perform the operation under test.

* Incorrect -

Check if the expected results have occurred.

* Incorrect -

Create an instance of the class under test.

* Incorrect -

Verify the state of the class under test.

* Incorrect -

I don't know yet.

3.Which situation would be a good use for a Spy in a mocking framework?

* Incorrect -

Mocking behavior of functional objects in the runtime

* Incorrect -

Uncovering bugs quickly

* Your choice: correct -

Recording interactions among objects

* Incorrect -

Quickly throwing an exception for methods

* Incorrect -

I don't know yet.

4.Given this failing test:

public class CalculatorTest {

public final void whenMoreThan2NumbersAreUsedThenExceptionIsThrown() {

Calculator c = new Calculator();

int result = c.sum(1,2);

assertEquals(3, result);

}

}

And the following requirements:

* The 'sum' method will return the sum of its parameters.
* Negative numbers will throw a runtime exception.
* Numbers bigger than 1,000,000,000 will throw a runtime exception.

Following the principles of test-driven development, which option would be the first version of the calculator implementation to make the above test pass?

* Incorrect -

public class Calculator {

public int sum(int a, int b) {

if(a >= 0 && b >= 0 && a <= 1\_000\_000\_000 && b <= 1\_000\_000\_000) {

return a + b;

} else {

return 0;

}

}

}

* Incorrect -

public class Calculator {

public int sum(int a, int b) {

return 0;

}

}

* Your choice: correct -

public class Calculator {

public int sum(int a, int b) {

return 3;

}

}

* Incorrect -

public class Calculator {

public int sum(int a, int b) {

if(a < 0 || b < 0) {

throw new RuntimeException("Negative numbers are not allowed");

} if(a > 1\_000\_000\_000 || b > 1\_000\_000\_000) {

throw new RuntimeException("Numbers bigger than 1,000,000,000 are not allowed");

}

return a + b;

}

}

* Incorrect -

I don't know yet.

5.In Cucumber, which annotation can be used to convert a given string to an object of a specific type?

* Incorrect -

@Convert

* Incorrect -

@Map

* Incorrect -

@AutoConvert

* Your choice: correct -

@Transform

* Incorrect -

I don't know yet.

6.Assume that you have to test a concrete class and you have to instantiate it in order to test. The constructor of that class needs an object of another interface. However creating an object of concrete implementation requires you to open a connection to a database. Which type of test double is best suited in this scenario to avoid having to connect to database?

* Correct -

Dummy

* Your choice: incorrect -

Mock

* Incorrect -

Spy

* Incorrect -

Stub

* Incorrect -

I don't know yet.

7.Considering this Checkstyle rule:

<module name="MagicNumber">

<property name="ignoreNumbers" value="0, 1"/>

</module>

And the following code snippet:

1. public class Check {

2. static final int SPECIAL\_NUMBER = 12 \* 3;

3. private int n = 7;

4.

5. void foo() {

6. int i = n + 1;

7. int j = i + 8;

8.

9. // ...

10. }

11. }

Which line numbers will be marked for a violation of the rule?

* Incorrect -

2, 3, 6, and 7

* Your choice: incorrect -

2, 3, and 7

* Incorrect -

3, 6, and 7

* Correct -

3 and 7

* Incorrect -

I don't know yet.

8.Which of the following options describes how some mutation testing tools work?

* Your choice: incorrect -

By using reflection to modify the runtime behavior of the program.

* Correct -

By manipulating the compiled bytecode with a library like ASM.

* Incorrect -

By using an aspect-oriented framework like AspectJ.

* Incorrect -

By modifying the classloader of the Java Virtual Machine.

* Incorrect -

I don't know yet.

9.Which code snippet accurately represents a Stub?

* Your choice: correct -

class FooDouble {

     public String bar() { return "baz"; }

}

* Incorrect -

class FooImplDouble implements FooInterface {}

* Incorrect -
* Incorrect -

class InMemoryUserDB implements DataBase {

private Map<String, User> users = new HashMap<String, User>();

public User get(String id) {

return users.get(id);

}

public Map<String, User> getAll() {

return users;

}

public User save(String id, User user) {

return users.put(id, user);

}

public User remove(String id){

return users.remove(id);

}

}

* Incorrect -

class PlainFileUserDB implements DataBase {

private CsvFileHelper db;

public CsvFileUserDB(CsvFileHelper db) {this.db=db;}

public User get(String id) {return db.get(id);}

public Map<String, User> getAll() {return db;}

public User save(String id, User user) {return db.put(id, user);}

public User remove(String id) {return db.remove(id);}

}

* Incorrect -

I don't know yet.

10.A passing test case is run with 8 mutations. The test fails with 4 of them, passes with 1, and can't be run because of compilation errors with 3.

Using this information, which of the following options state the correct number or killed mutations and whether the test case is good or not?

* Correct -

Four of five valid mutations are killed. It can be considered a good test case.

* Incorrect -

Three of eight valid mutations are killed. It can be considered a bad test case.

* Your choice: incorrect -

One of five valid mutations is killed. It can be considered a good test case.

* Incorrect -

Seven of eight valid mutations are killed. It can be considered a bad test case.

* Incorrect -

I don't know yet.

11.Given:

int n = 5;

if ( n >= 3 ) {

return 1;

} else {

return 0;

}

Which of the following options is an equivalent mutation?

* Incorrect -

int n = 5; if ( n >= 3 ) { return 0; } else { return 1; }

* Your choice: correct -

int n = 5; if ( n > 3 ) { return 1; } else { return 0; }

* Incorrect -

int n = 0; if ( n >= 3 ) { return 1; } else { return 0; }

* Incorrect -

int n = 5; if ( n < 3 ) { return 1; } else { return 0; }

* Incorrect -

I don't know yet.

12.

What is the result of the following test?

import static org.hamcrest.MatcherAssert.assertThat;

import static org.hamcrest.Matchers.\*;

public class MyTest {

@Test

public testList() {

List<Integer> list = Arrays.asList(1, 2, 3, 4);

assertThat(list, both(hasSize(4)).and(contains(2, 4, 1)));

}

}

* Correct -

The test fails because the actual list (1, 2, 3, 4) doesn't match the expected list (2, 4, 1).

* Your choice: incorrect -

The test passes because the list has a size of 4 and contains the specified elements.

* Incorrect -

The test passes because the list matches the first matcher (hasSize(4)). The result of the second matcher doesn't matter.

* Incorrect -

The test fails because the list doesn't contain the elements 2, 4, 1 in that order.

* Incorrect -

I don't know yet.

13.Which of the following definitions closely depicts a stub?

* Incorrect -

An object that acts as a broker between the test and mocked object and transfers the data to and from it.

* Incorrect -

An object used during a test, which holds the argument data required for calling a method.

* Incorrect -

An object that simply ignores the calls made to its methods.

* Your choice: correct -

An object that has canned responses to method calls during a test.

* Incorrect -

I don't know yet.

14.

Which problem can be solved using dependency injection?

* Incorrect -

Testing prototype and other throw-away investigative pieces of code

* Incorrect -

Having to pass any number of parameters to a method

* Incorrect -

Defining how an object may use another object

* Your choice: correct -

Supporting different configurations at runtime

* Incorrect -

I don't know yet.

15.

Which of the following statements about triangulation is TRUE?

* Incorrect -

Triangulation is a core technique of behavior-driven development.

* Incorrect -

Using triangulation, you type the real implementation of the operation under test first.

* Incorrect -

Using triangulation, you can easily find a bug in a piece of code.

* Correct -

Triangulation is an alternative to the "Fake It" technique.

* Incorrect -

I don't know yet.

16.

Which Selenium code snippet is a correct proxy for a ChromeDriver?

* Correct -

ChromeOptions options = new ChromeOptions();

Proxy proxy = new Proxy();

proxy.setHttpProxy("myproxy:4000");

options.setCapability("proxy", proxy);

ChromeDriver driver = new ChromeDriver(options);

* Incorrect -

ChromeOptions options = new ChromeOptions();

options.setCapability("proxy", "myproxy:4000");

ChromeDriver driver = new ChromeDriver(options);

* Your choice: incorrect -

ChromeOptions options = new ChromeOptions();

options.enableProxies();

Proxy proxy = new Proxy();

proxy.setHttpProxy("myproxy:4000");

ChromeDriver driver = new ChromeDriver(options);

driver.setCapability("proxy", proxy);

* Incorrect -

ChromeDriver driver = new ChromeDriver();

driver.setProxy("myproxy:4000");

* Incorrect -

I don't know yet.

17.

You must build integration tests for an e-commerce system that is primarily accessed through a user interface built for customers. There is also a portal that provides scaffolding for developers to test functionality. In the backend, there is a service layer that provides data to the client. And finally, there is a database that holds state for the application. Which layer is the least suited to integration testing?

* Your choice: correct -

The developer portal

* Incorrect -

The service layer

* Incorrect -

The main user interface

* Incorrect -

The database layer

* Incorrect -

I don't know yet.

18.

Which of the following Selenium functions performs a right click on the element?

* Incorrect -

click(element, false);

* Your choice: incorrect -

rightClick(element);

* Incorrect -

click(element, BUTTON.RIGHT);

* Correct -

contextClick(element);

* Incorrect -

I don't know yet.

1.

Which is an example of the problems that can be found using static analysis?

* Your choice: correct -

Unnecessary object creation

* Incorrect -

Low quality unit tests

* Incorrect -

Incompatibilities between Java versions

* Incorrect -

Detection of a memory leak at runtime

* Incorrect -

I don't know yet.

2.

Which option is a common refactoring?

* Your choice: correct -

Pull up method

* Incorrect -

Template method

* Incorrect -

Value object

* Incorrect -

Factory method

* Incorrect -

I don't know yet.

3.

Given:

public class InvoiceService {

private InvoiceRepository repository;

public InvoiceService(InvoiceRepository repository) {

this.repository = repository;

}

public Double calculateTotal(Invoice i) {

// ...

}

public void generateReport(Sale s, Invoice i) {

// ...

}

public Double calculateDiscount(Sale s) {

// ...

}

public Invoice generateInvoice(Sale s) {

// ...

}

private Double getExchangeRate() {

// ...

}

//...

}

Which action will make this class more testable?

* Your choice: correct -

Modify it so it doesn't use a repository class.

* Incorrect -

Modify it so it can perform more tasks.

* Incorrect -

Modify it so it has more private methods.

* Incorrect -

Modify it so it implements an interface.

* Incorrect -

I don't know yet.

4.

Using Selenium, how would you decorate an element when we need to initialize an element whose name is "textfield" and class is "blue"?

* Incorrect -

@FindBys( [ @FindBy(how = How.NAME, using = "textfield"), @FindBy(how = How.CLASS\_NAME, using = "blue") ] ) private WebElement blueTextField;

* Your choice: correct -

@FindBys( { @FindBy(how = How.NAME, using = "textfield"), @FindBy(how = How.CLASS\_NAME, using = "blue") } ) private WebElement blueTextField;

* Incorrect -

@FindBy([(how = How.NAME, using = "textfield"),(how = How.CLASS\_NAME, using = "blue")) private WebElement blueTextField;

* Incorrect -

@FindAlls({ @FindBy(how = How.NAME, using = "textfield"), @FindBy(how = How.CLASS\_NAME, using = "blue") }) private WebElement blueTextField;

* Incorrect -

I don't know yet.

5.

Have a look at this scenario step.

Then get book list response should contain records as

| isbn | title | price |

| I2389423 | Cucumber for Java | 35.0 |

| I8459827 | Mastering Gherkin | 55.0 |

You have a class named book of following structure.

public class Book {

public String isbn;

public String title;

public String price;

}

Which of the following code snippets gets the data in code effectively?

* Incorrect -

@Then("^the book list response should contain records as$")

public void validateBooks(Book[] booksList) {

// test code

}

* Incorrect -

@Then("^the book list response should contain records as$")

public void validateBooks(DataTable table) {

List<Book> booksList = table.asList(Book.class);

// test code

}

* Correct -

@Then("^the book list response should contain records as$")

public void validateBooks(List<Book> booksList) {

// test code

}

* Your choice: incorrect -

@Then("^the book list response should contain records as$")

public void validateBooks(DataTable table) {

List<List<String>> data = table.raw();

// test code

}

* Incorrect -

I don't know yet.

6.

Which preference can be used to set the home page for FirefoxDriver?

* Your choice: correct -

browser.startup.homepage

* Incorrect -

firefox.homepage

* Incorrect -

firefox.startup.homepage

* Incorrect -

browser.homepage

* Incorrect -

I don't know yet.

7.

Which of the following are principles of behavior-driven development (BDD)?

1. 1. Business and Technology should refer to the same system in the same way
2. 2. Complete automation of all tests can be achieved through effective use of BDD
3. 3. Up-front analysis, design and planning all have a diminishing return

* Incorrect -

1, 2, & 3

* Your choice: incorrect -

1 & 2

* Incorrect -

2 & 3

* Correct -

1 & 3

* Incorrect -

I don't know yet.

8.

Considering this Checkstyle rule:

<module name="MagicNumber">

<property name="ignoreNumbers" value="0, 1"/>

</module>

And the following code snippet:

1. public class Check {

2. static final int SPECIAL\_NUMBER = 12 \* 3;

3. private int n = 7;

4.

5. void foo() {

6. int i = n + 1;

7. int j = i + 8;

8.

9. // ...

10. }

11. }

Which line numbers will be marked for a violation of the rule?

* Your choice: correct -

3 and 7

* Incorrect -

3, 6, and 7

* Incorrect -

2, 3, 6, and 7

* Incorrect -

2, 3, and 7

* Incorrect -

I don't know yet.

9.

Using triangulation, which of the options represent a way to implement the following requirement?

Write a program that returns an array of integers from 1 to N omitting even numbers.

* Incorrect -

1. Write a test for the sequence 1 to 1. Change the implementation to make the test pass.

2. Refactor if necessary.

3. Write a test for the sequence 1 to 10. Change the implementation to make the test pass.

4. Refactor if necessary.

5. Write a test for the sequence 1 to 50. Change the implementation to make the test pass.

6. Refactor if necessary.

* Incorrect -

1. Write a test for the sequence 1 to 1. Hard-code the solution to make the test pass.

2. Write a test for the sequence 1 to 2. Change the implementation to make the test pass.

3. Write a test for the sequence 1 to 3. Change the implementation to make the test pass.

* Your choice: correct -

1. Write a test for the sequence 1 to 1. Hard-code the solution to make the test pass.

2. Write a test for the sequence 1 to 2. Change the implementation to make the test pass.

3. Refactor if necessary.

4. Write a test for the sequence 1 to 3. Change the implementation to make the test pass.

5. Refactor if necessary.

6. Write a test for the sequence 1 to 4. Change the implementation to make the test pass.

7. Refactor if necessary.

* Incorrect -

1. Write a test for the sequence 1 to 4. Code the implementation that makes the test pass.

2. Write a test for the sequence 1 to 99. Change the implementation to make the test pass.

3. Refactor if necessary.

* Incorrect -

I don't know yet.

10.

What's another name for the London style test-driven development?

* Incorrect -

Inside-Out

* Your choice: correct -

Mockist

* Incorrect -

Classic

* Incorrect -

Bottom-up

* Incorrect -

I don't know yet.

11.

Which of the following is one type of mutation operator to avoid?

* Incorrect -

Operators that are subjected to the test data set.

* Incorrect -

Operators that do the opposite thing as the original program.

* Your choice: correct -

Operators that are syntactically incorrect.

* Incorrect -

Operators that are hard to kill.

* Incorrect -

I don't know yet.

12.

In JUnit 5, which of the following methods is used to assert that execution of a method completes before the given timeout, aborting the execution if the timeout is exceeded?

* Incorrect -

assertTimeout

* Your choice: correct -

assertTimeoutPreemptively

* Incorrect -

assertTimeoutAborting

* Incorrect -

assertTimeoutExceeding

* Incorrect -

I don't know yet.

13.

How do you configure a database to be in a clean state every time you run Integration tests using Hibernate? Clean state means that the database doesn't contain any data.

* Your choice: correct -

<property name="hibernate.hbm2ddl.auto" value="create"/>

* Incorrect -

<property name="hibernate.hbm2ddl.cleanup" value="always"/>

* Incorrect -

<property name="hibernate.hbm2ddl.auto" value="drop-create"/>

* Incorrect -

<property name="hibernate.hbm2ddl.auto" value="drop"/>

<property name="hibernate.hbm2ddl.auto" value="create"/>

* Incorrect -

I don't know yet.

14.

In Cucumber, which attribute can provide configuration through a JSON file?

* Incorrect -

-jsonConfig

* Incorrect -

-config

* Correct -

-nodeConfig

* Incorrect -

-configuration

* Incorrect -

I don't know yet.

15.

Which additional dependency is required for Mockito to mock all the methods in the following code snippet?

class ClassToBeTested { public void methodToBeTested() { // code under test doSomething(); } public final void doSomething() { // code to be tested or gets called while testing other methods throw new TestBlockerException("Final methods cannot be overridden"); } }

* Your choice: incorrect -

org.mockito:mock-plus

* Correct -

org.mockito:mockito-inline

* Incorrect -

org.mockito:mockito-final

* Incorrect -

No additional dependency is necessary.

* Incorrect -

I don't know yet.

16.

Which Selenium code snippet is a correct proxy for a ChromeDriver?

* Incorrect -

ChromeOptions options = new ChromeOptions();

options.enableProxies();

Proxy proxy = new Proxy();

proxy.setHttpProxy("myproxy:4000");

ChromeDriver driver = new ChromeDriver(options);

driver.setCapability("proxy", proxy);

* Incorrect -

ChromeDriver driver = new ChromeDriver();

driver.setProxy("myproxy:4000");

* Correct -

ChromeOptions options = new ChromeOptions();

Proxy proxy = new Proxy();

proxy.setHttpProxy("myproxy:4000");

options.setCapability("proxy", proxy);

ChromeDriver driver = new ChromeDriver(options);

* Your choice: incorrect -

ChromeOptions options = new ChromeOptions();

options.setCapability("proxy", "myproxy:4000");

ChromeDriver driver = new ChromeDriver(options);

* Incorrect -

I don't know yet.

17.

What is the result of the following test?

import static org.hamcrest.MatcherAssert.assertThat;

import static org.hamcrest.Matchers.\*;

public class MyTest {

@Test

public testList() {

List<Integer> list = Arrays.asList(1, 2, 3, 4);

assertThat(list, both(hasSize(4)).and(contains(2, 4, 1)));

}

}

* Your choice: correct -

The test fails because the actual list (1, 2, 3, 4) doesn't match the expected list (2, 4, 1).

* Incorrect -

The test fails because the list doesn't contain the elements 2, 4, 1 in that order.

* Incorrect -

The test passes because the list matches the first matcher (hasSize(4)). The result of the second matcher doesn't matter.

* Incorrect -

The test passes because the list has a size of 4 and contains the specified elements.

* Incorrect -

I don't know yet.

18.

Assume that you have to test a concrete class and you have to instantiate it in order to test. The constructor of that class needs an object of another interface. However creating an object of concrete implementation requires you to open a connection to a database. Which type of test double is best suited in this scenario to avoid having to connect to database?

* Incorrect -

Mock

* Your choice: correct -

Dummy

* Incorrect -

Spy

* Incorrect -

Stub

* Incorrect -

I don't know yet.

1.

In JUnit 5, which method can you use to set up an expensive operation that is going to be used within a test class?

* Incorrect -

A method annotated with @BeforeEach

* Correct -

A method annotated with @BeforeAll

* Incorrect -

The constructor of the test class

* Your choice: incorrect -

A method annotated with @Test

2.

What action can you take to make a unit test faster?

* Incorrect -

Don't test the slow parts of the application.

* Incorrect -

Run the unit test in a continuous integration server.

* Your choice: incorrect -

Build and run the application code and the test code separately.

* Correct -

Mock external dependencies like a database.

3.

What does "Asserting first" mean?

* Incorrect -

When creating a test, determine the assertion method you'll use first.

* Your choice: incorrect -

The assertion part of the test is the most important part.

* Correct -

When creating a test, determine the expected result first.

* Incorrect -

A test method should start with an assert statement.

4.

Why should we only test the behavior, but not the implementation?

* Incorrect -

It takes too long to run the tests.

* Your choice: correct -

Tests will then be too brittle. Any small change in implementation breaks tests.

* Incorrect -

Behavior will never change.

* Incorrect -

We should always test the implementation.

5.

In the following instance, what is the purpose of the @Primary annotation?

@Profile("test")

@Configuration

class TestConfiguration {

@Bean

@Primary

public UserManager userService() {

return Mockito.mock(UserManager.class);

}

}

* Correct -

@Primary forces the Spring framework to return this instance when used in the 'test' profile.

* Incorrect -

@Primary forces the Spring framework to return this instance when used in every profile.

* Incorrect -

@Primary causes UserManager to be instantiated like a Singleton.

* Your choice: incorrect -

@Primary causes UserManager to be instantiated like an interface.

6.

Which of the following options best defines the goal of a test suite?

* Your choice: correct -

It groups similar test cases together.

* Incorrect -

It tests the UI of the application.

* Incorrect -

It tests a particular path of a business rule.

* Incorrect -

It allows you to filter test cases.

7.

Which Selenium WebDriver implementation allows you to change the network connection type to ConnectionType.AIRPLANE\_MODE?

* Incorrect -

InternetExplorerDriver

* Incorrect -

FirefoxDriver

* Your choice: correct -

ChromeDriver

* Incorrect -

EdgeDriver

8.

Consider the following test class:

public class UserTest { private String username; private String password; private Integer userId; private ISerializer serializer; void testValidUsername() { // ... } void testValidPassword() { // ... } void testUserCreation() { // ... } void testUserConfigurationAfterRetrieval() { // ... } }

Which attribute is the most likely candidate for dependency injection?

* Incorrect -

user

* Incorrect -

password

* Your choice: correct -

serializer

* Incorrect -

username

9.

Given that your Java application code resides in the directory src/com/example/app/web, which of the following is a valid example of organizing the unit tests for this application code?

* Incorrect -

Put the test classes in the directory src/com/example/app/web/unit-tests under the same project.

* Incorrect -

Put the test classes in a different project with the same directory structure.

* Incorrect -

Put the test classes in a different project with a different directory structure.

* Your choice: correct -

Put the test classes in the directory test/com/example/app/web under the same project.

10.

In TestNG, what's the meaning of annotating a test method with:

@Test(expectedExceptions = ArithmeticException.class)

* Correct -

The test method is expected to throw an ArithmeticException. If a different one is thrown, the test will fail. If none is thrown, the test will also fail.

* Your choice: incorrect -

The test method could throw an ArithmeticException. If a different one is thrown, the test could pass. If none is thrown, the test could still pass.

* Incorrect -

The test method will catch any ArithmeticException unexpectedly thrown.

* Incorrect -

The test method is expected to throw an ArithmeticException. If a different one is thrown, the test will fail. If none is thrown, the test could still pass.

11.

Which of the following statements describes how dependency injection can be applied to testing?

* Incorrect -

To expose private methods so they can be tested.

* Correct -

To inject a fake object instead of the real implementation of a complex object.

* Your choice: incorrect -

To inject a test class into a test suite and create dynamic suites.

* Incorrect -

To implement the Service Locator pattern and abstract the process of getting an object.

12.

Consider a Java-based service that calculates the discount that should be applied according to the following rules:

- A purchase in the range of $1 to $49.99 has 10% discount.

- A purchase in the range of $50 to $99.99 has 20% discount.

- A purchase of $100 or more has 30% discount.

According to boundary-value analysis, which of the following options is an example of the minimum set of values you should test?

* Incorrect -

1, 49.99, 50, 99.99, 100, 101.

* Your choice: correct -

0.99, 1, 49.99, 50, 99.99, 100.

* Incorrect -

1, 50, 100.

* Incorrect -

1, 49.99, 50, 99.99, 101.

13.

Which of the following frameworks can be used to write integration tests for controllers without using web interface?

* Your choice: incorrect -

JMeter

* Incorrect -

Selenium

* Incorrect -

Geb

* Correct -

MockMVC

14.

What is the most important aspect in behavior-driven development?

* Correct -

Collaboration between business experts, developers, and testers while writing features.

* Incorrect -

Having the application mocks ready before starting the development.

* Incorrect -

Being able to document everything as requirements before starting development of application.

* Your choice: incorrect -

Finishing the application before starting to write tests in BDD.

15.

Knowing that you can use XPath results matcher to validate responses in MockMVC, which of the following statements is true?

* Incorrect -

You should use it as much as possible as this will give complete coverage on UI structure as well.

* Your choice: correct -

It is better to avoid it unless it is extremely necessary as it will make the tests too fragile owing to the fact that UI changes frequently.

* Incorrect -

It is better to avoid it given the fact that it cannot validate nested elements.

* Incorrect -

You should not use it at all because you will never have to test UI elements structure in an Integration test.

16.

What is the result of the following test?

@Test

public void spytest() {

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

List<String> listSpy = Mockito.spy(list);

listSpy.add("one");

listSpy.add("two");

assertEquals(2, listSpy.size());

}

* Incorrect -

Test fails with a Null Pointer Exception.

* Incorrect -

Test Fails with Assertion Exception.

* Correct -

Test passes.

* Your choice: incorrect -

Test fails with behavior not defined exception.

17.

What JUnit 5 tests does the following Gradle configuration run?

junitPlatform {

// ...

filters {

tags {

include 'services & feature-2'

}

packages {

include 'com.example.product'

}

includeClassNamePattern '.\*Spec'

}

// ...

}

* Your choice: correct -

It will run tests whose class names end in 'Spec', and are located in 'com.example.product', and are tagged with 'services' and 'feature-2'.

* Incorrect -

It will run tests whose class names end in 'Spec', or are located in 'com.example.product', or are tagged with 'services' and 'feature-2'.

* Incorrect -

It will run tests whose class names end in 'Spec', and are located in 'com.example.product', and are tagged with either 'services' or 'feature-2'.

* Incorrect -

It will run tests whose class names end in 'Spec', or are located in 'com.example.product', or are tagged with either 'services' or 'feature-2'.

18.

Consider the following workflow in an e-commerce Java system:

- Reading a single JSON file from a FTP server to get the product sale data

- Parsing the JSON data using an established third-party library

- Calculating the product sale statistics

- Saving the results to the database

Which step(s) will involve the most unit testing of actual logic, rather than mocking?

* Correct -

Calculating the product sale statistics.

* Your choice: incorrect -

From reading the JSON file to parsing the data.

* Incorrect -

Saving the results to the database.

* Incorrect -

Reading the JSON file.

1. Why is HSQLDB a suitable choice for integration testing?

Incorrect -It is free of charge, when used in a testing context.

Incorrect -Its SQL syntax is simpler than the syntax of other proprietary DBs.

Your choice: correct -It runs in memory and can be used as a replacement for a standalone DB.

2. Why is mutation testing considered a form of white-box testing?

Incorrect -Because it cannot be done without an automation tool.

Incorrect -Because it doesn't involve changes to the source code.

Incorrect -Because it is costly and time consuming.

Your choice: correct -Because it involves changes to the source code.

3. Given that you just wrote the following failing test, which is asserting valid behavior:

public class TestPhone {

  @Test

  void testValidPhoneNumberLength() {

    Phone phone = new Phone();

    assertTrue(phone.validate("55-98670011"));

  }

}

In Test-driven Development, what's the next step?

Incorrect -Refactor the test to improve the code.

Incorrect -Write a specification for that test.

Incorrect -Refactor the application to improve the code.

Your choice: correct -Write the smallest amount of code to make the test pass.

4. In JUnit 5, which method can you use to signal the intention of executing it before the other tests in the current test class?

Incorrect -The constructor of the test class

Your choice: incorrect -A method annotated with @Test

Incorrect -A method annotated with @BeforeEach

Correct -A method annotated with @BeforeAll

5. Which of the following tools helps programmers write Java code that adheres to a coding standard?

Incorrect -JSHint

Correct -Checkstyle

6. What is the result of the following test?

@Test

    public void spytest() {

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

        List<String> listSpy = Mockito.spy(list);

        listSpy.add("one");

        listSpy.add("two");

        assertEquals(2, listSpy.size());

    }

Incorrect -Test fails with behavior not defined exception.

Incorrect -Test Fails with Assertion Exception.

Your choice: correct -Test passes.

7. In your Java test class, which is using JUnit5, all your test methods use a mock of a repository class that needs to be initialized every time a test is executed. Which of the following options describe the best way to set up the repository mock?

Incorrect -Initialize the repository mock in a method annotated with @BeforeAll.

Incorrect -Initialize the repository mock in a method annotated with @AfterEach.

Incorrect -Initialize the repository mock inside all test methods.

Your choice: correct -Initialize the repository mock in a method annotated with @BeforeEach.

8. Which action corresponds to the "Red" part in the so-called Red-Green-Refactor cycle?

Incorrect -Writing a test with mocks or fake objects

Correct -Writing a test that fails

9. Which of the following options could be applied to reduce the cyclomatic complexity of a piece of code?

Incorrect -Reducing the number of methods in a class.

Incorrect -Applying a naming convention.

Correct -Avoid writing if statements with many branches.

Question was not answered in the time given. Recorded as incorrect.

1. Given:

public class MyTest {

  @Test

  public void testDefaultName() {

    Person p = new Person();

    assertEquals("John Doe", p.getName(), "The default name is not set correctly");

  }

}

What happens when the assertion fails?

Incorrect -The test is NOT marked as failed but the specified message is shown in the output.

Incorrect -The test is marked as failed but only the specified message is shown in the output.

Correct -An exception of type AssertionError (or a subclass) is thrown with the specified message and shown in the output.

Incorrect -An exception of type AssertionError (or a subclass) is thrown. The specified message is shown only in the test report (if generated).

Incorrect -I don't know yet.

2. In your Java test class, which is using JUnit5, all your test methods use a mock of a repository class that needs to be initialized every time a test is executed. Which of the following options describe the best way to set up the repository mock?

Incorrect -Initialize the repository mock in a method annotated with @AfterEach.

Incorrect -Initialize the repository mock inside all test methods.

Correct -Initialize the repository mock in a method annotated with @BeforeEach.

Your choice: incorrect -Initialize the repository mock in a method annotated with @BeforeAll.

Incorrect -I don't know yet.

3. How can you apply the following rule of Test-driven Development? Only write code to fix a failing test.

Incorrect -By writing the code first, then writing a failing test, and then fix the test.

Incorrect -By writing a test but never allowing it to fail.

Your choice: correct -By writing a failing test first, and then writing the code that makes it pass.

Incorrect -By writing all the tests of the application first, and then writing the application code.

Incorrect -I don't know yet.

4. Which statement is true about writing unit tests?

Incorrect -In addition to your application code, you should also test external libraries and services.

Incorrect -A unit test should only focus on edge cases.

Correct -A unit test should test only one thing.

Your choice: incorrect -Code coverage must be 90% or higher for tests to execute.

Incorrect -I don't know yet.

5. Which of the following frameworks can be used to write integration tests for controllers without using web interface?

Your choice: correct -MockMVC

Incorrect -Geb

Incorrect -JMeter

Incorrect -Selenium

Incorrect -I don't know yet.

6. Given:

package com.example;

import org.testng.annotations.Test;

public class InvoiceTest {

        @Test(groups={"fast", "invoice"})

        public void testGenerateInvoice() {

          System.out.println("testGenerateInvoice");

        }

        @Test(groups={"fast"})

        public void testGenerateEmptyInvoice() {

          System.out.println("testGenerateEmptyInvoice");

        }

        @Test(groups="invoice")

        public void testUpdateInvoice() {

                System.out.println("testUpdateInvoice");

        }

        @Test(groups="fast")

        public void testFindInvoice() {

                System.out.println("testFindInvoice");

        }

}

And the configuration:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="Suite">

  <test name="invoice">

        <groups>

      <run>

        <include name="fast"/>

      </run>

    </groups>

    <classes>

      <class name="com.example.InvoiceTest"/>

    </classes>

  </test>

</suite>

What is the output when running the test with the above configuration?

Your choice: incorrect -testGenerateInvoice

testGenerateEmptyInvoice

testUpdateInvoice

testFindInvoice

Incorrect -testGenerateEmptyInvoice

testFindInvoice

Correct -testGenerateInvoice

testGenerateEmptyInvoice

testFindInvoice

Incorrect -testGenerateEmptyInvoice

testUpdateInvoice

testFindInvoice

Incorrect -I don't know yet.

7. Given the following JUnit5 test:

@Test

void testPerson() {

  Person person = new Woman();

  person.setAge(25);

  assertTrue(person.isAdult() && !person.isMan(), "Person must be adult and not man");

}

The following options show the above test method after refactoring has been applied. Which one improved it the most?

Correct -

@Test

void shouldBeAnAdultNotManWhenWoman25yr(){

  Person person = new Woman();

  person.setAge(25);

  assertTrue(person.isAdult(), "Person must be an adult");

  assertFalse(person.isMan(), "Person must not be a man");

}

Incorrect -

@Test

void testPerson(){

  Person person = factory.getPerson(Gender.Woman);

  person.setAge(25);

  assertEquals(true, person.isAdult() && person.isWoman(), "Person must be adult and woman");

}

Incorrect -

@Test

void testPerson(){

  Person person = new Woman();

  person.setAge(Constants.AGE\_25);

  assertEquals(true, person.isAdult() && !person.isMan(), "Person must be adult and not man");

}

Your choice: incorrect -

@Test

void testPerson(){

  Person person = new Woman();

  person.setAge(25);

  assertEquals(true, person.isAdult() && !person.isMan(), "Person must be adult and not man");

}

Incorrect -I don't know yet.

8. What is wrong with the following JUnit5 test class?

public class AccountTest {

  private Account account;

  @BeforeAll

  void init() {

    this.account = new Account();

  }

  @Test

  void testDeposit() {

    this.account.deposit(1000);

    assertEquals(1000, this.account.getBalance());

    assertEquals(1, this.account.getNumberOfTransactions());

  }

  @Test

  void testWithdrawal() {

    this.account.withdrawal(200);

    assertEquals(800, this.account.getBalance());

    assertEquals(2, this.account.getNumberOfTransactions());

  }

}

Incorrect -The creation of the Account instance should be inside the test methods.

Incorrect -The class doesn't use dependency injection for the Account instance.

Incorrect -The tests have more than one assert method.

Your choice: correct -The second test depends on the other. Unit tests should work independently.

Incorrect -I don't know yet.

9. Given:List<Integer> list = Arrays.asList(1, 2, 3, 4);Using Hamcrest, how would you test the list that contains the element 3?

Your choice: incorrect -assertThat(list, hasValue(3));

Incorrect -assertThat(list, in(3));

Correct -assertThat(list, hasItem(3));

Incorrect -assertThat(list, everyItem(equalsTo(3)));

Incorrect -I don't know yet.

Question was not answered in the time given. Recorded as incorrect.

10. Which of the following statements describes how dependency injection can be applied to testing?

Incorrect -To expose private methods so they can be tested.

Incorrect -To implement the Service Locator pattern and abstract the process of getting an object.

Correct -To inject a fake object instead of the real implementation of a complex object.

Incorrect -To inject a test class into a test suite and create dynamic suites.

Incorrect -I don't know yet.

11. What is a principle of test-driven development?

Your choice: correct -Write a test and then write enough code so that the test passes.

Incorrect -Refactor application code after a failing test.

Incorrect -Mock all external dependencies in unit tests.

Incorrect -Write detailed specifications from which you can derive tests.

Incorrect -I don't know yet.

12. Consider the following workflow in an e-commerce Java system: - Reading a single JSON file from a FTP server to get the product sale data

- Parsing the JSON data using an established third-party library

- Calculating the product sale statistics

- Saving the results to the database

Which step(s) will involve the most unit testing of actual logic, rather than mocking?

Your choice: incorrect -From reading the JSON file to parsing the data.

Incorrect -Saving the results to the database.

Incorrect -Reading the JSON file.

Correct -Calculating the product sale statistics.

Incorrect -I don't know yet.

13. Which of the following options could be applied to reduce the cyclomatic complexity of a piece of code?

Correct -Avoid writing if statements with many branches.

Incorrect -Applying a naming convention.

Incorrect -Reducing the number of methods in a class.

Your choice: incorrect -Using unchecked exceptions instead of checked exceptions.

Incorrect -I don't know yet.

14. Consider a Java-based service that calculates the discount that should be applied according to the following rules: - A purchase in the range of $1 to $49.99 has 10% discount.

- A purchase in the range of $50 to $99.99 has 20% discount.

- A purchase of $100 or more has 30% discount.

According to boundary-value analysis, which of the following options is an example of the minimum set of values you should test?

Incorrect -1, 49.99, 50, 99.99, 101.

Incorrect -1, 50, 100.

Your choice: correct -0.99, 1, 49.99, 50, 99.99, 100.

Incorrect -1, 49.99, 50, 99.99, 100, 101.

Incorrect -I don't know yet.

15. In TestNG, what's the meaning of annotating a test method with: @Test(expectedExceptions = ArithmeticException.class)

Incorrect -The test method could throw an ArithmeticException. If a different one is thrown, the test could pass. If none is thrown, the test could still pass.

Incorrect -The test method will catch any ArithmeticException unexpectedly thrown.

Your choice: incorrect -The test method is expected to throw an ArithmeticException. If a different one is thrown, the test will fail. If none is thrown, the test could still pass.

Correct -The test method is expected to throw an ArithmeticException. If a different one is thrown, the test will fail. If none is thrown, the test will also fail.

Incorrect -I don't know yet.

16. In the following instance, what is the purpose of the @Primary annotation?

@Profile("test")

@Configuration

class TestConfiguration {

    @Bean

    @Primary

    public UserManager userService() {

        return Mockito.mock(UserManager.class);

    }

}

Correct -@Primary forces the Spring framework to return this instance when used in the 'test' profile.

Incorrect -@Primary causes UserManager to be instantiated like a Singleton.

Incorrect -@Primary causes UserManager to be instantiated like an interface.

Your choice: incorrect -@Primary forces the Spring framework to return this instance when used in every profile.

Incorrect -I don't know yet.

17. In Gherkin, what is each line of a scenario is called?

Your choice: correct -Step

Incorrect -Section

Incorrect -Pace

Incorrect -Instruction

Incorrect -I don't know yet.

18. In an ideal testing pyramid, which layer should contain the highest coverage and lowest coverage?

Incorrect -User interface (UI) automated testing - highestUnit testing - lowest

Correct -Unit testing - highestUser interface (UI) automated testing - lowest

Your choice: incorrect -User interface (UI) automated testing - highest

Integration testing - lowest

Incorrect -Integration testing - highest

Unit testing - lowest

Incorrect -I don't know yet.

19.Which question should you ask yourself when deciding what to test?

* Incorrect -

Does the architecture of the application make it testable?

* Incorrect -

Should I test all the private methods of this class?

* Incorrect -

Which testing framework should I use?

* Your choice: correct -

What is the expected output when performing X?

* Incorrect -

I don't know yet.

20. Which option represents a source for parameterized tests in JUnit 5?

* Incorrect -

An Excel (.xls) file.

* Incorrect -

A Word (.doc) file.

* Your choice: correct -

A CSV (.csv) file.

* Incorrect -

An HTML (.html) file.

* Incorrect -

I don't know yet.

21.What does the following JaCoCo Maven configuration do?

<execution>

<id>jacoco-check</id>

<goals>

<goal>check</goal>

</goals>

<configuration>

<rules>

<rule>

<element>CLASS</element>

<limits>

<limit>

<counter>LINE</counter>

<value>COVEREDRATIO</value>

<minimum>0.60</minimum>

</limit>

</limits>

</rule>

</rules>

</configuration>

</execution>

* Incorrect -

Shows a warning if a minimum line coverage of 60% for the whole application is not met

* Correct -

Requires a minimum line coverage of 60% for every class, otherwise the build will fail

* Your choice: incorrect -

Requires a minimum line coverage of 60% for the whole application, otherwise, the build will fail

* Incorrect -

Shows a warning if a minimum line coverage of 60% for every class is not met

* Incorrect -

I don't know yet.

21.You are using mutation testing but the process is taking too long to run. What can you do with a mutation testing tool to speed things up?

* Incorrect -

Change your testing framework, for example, use TestNG instead of JUnit.

* Incorrect -

Configure the way in which mutants are inserted.

* Your choice: incorrect -

Compile the code before running the mutation testing tool.

* Correct -

Target only those packages or classes that are currently of interest.

* Incorrect -

I don't know yet.

22.Which component of Selenium Grid gives you information about the Selenium Server and the browser instances connected to it?

* Correct -

Selenium Grid Hub

* Your choice: incorrect -

Selenium IDE

* Incorrect -

Selenium Remote Control

* Incorrect -

Selenium Grid Node

* Incorrect -

I don't know yet.

23.Which code snippet tests an EJB appropriately using Arquillian?

* Incorrect -

private CaseService caseService = new CaseService();

@Test

public void convertToLowerCase(){

assertEquals("seattle", caseService.toLowerCase("Seattle"));

}

* Correct -

@Inject

private CaseService caseService;

@Test

public void convertToLowerCase(){

assertEquals("seattle", caseService.toLowerCase("Seattle"));

}

* Incorrect -

@Test

public void convertToLowerCase(){

CaseService caseService = new CaseService();

assertEquals("seattle", caseService.toLowerCase("Seattle"));

}

* Your choice: incorrect -

@Test

public void convertToLowerCase(){

@Inject private CaseService caseService;

assertEquals("seattle", caseService.toLowerCase("Seattle"));

}

* Incorrect -

I don't know yet.

24.Which statement best describes what the term "data-driven testing" means?

* Correct -

Using a table of data with input and verifying values instead of hard-coded values in tests.

* Incorrect -

Loading complete test cases from Excel files or database tables into a testing framework.

* Your choice: incorrect -

Designing the database model first and then the test cases and the application.

* Incorrect -

Testing if the data contained in a database table is saved correctly.

* Incorrect -

I don't know yet.

25.What's the definition of the mutation score in mutation testing?

* Correct -

The percentage of killed mutants and total the number of mutants.

* Incorrect -

The number of mutations executed.

* Your choice: incorrect -

The number of killed mutants.

* Incorrect -

The percentage of failed tests and the total number of tests.

* Incorrect -

I don't know yet.

26. Using JUnit 5, how would you test that the execution of the method calculateSales() completes in less than 1 minute?

* Correct -

@Test void timeout() { assertTimeout(Duration.ofMinutes(1), () -> { calculateSales(); }); }

* Your choice: incorrect -

@Test(timeout=1000) void timeout() { calculateSales(); }

* Incorrect -

@Test void timeout() { assertTimeout(1, calculateSales()); }

* Incorrect -

@Test(timeout=1) void timeout() { calculateSales(); }

* Incorrect -

I don't know yet.

27.Which of the following is a characteristic of a good unit test?

* Incorrect -

It only tests error or exceptional cases.

* Your choice: incorrect -

It's written in a thread-safe way.

* Correct -

It is deterministic.

* Incorrect -

It only has one assert method.

* Incorrect -

I don't know yet.

28. Which of the following is an example of an SQL database which offers an explicit in-memory mode of operation?

* Incorrect -

Neo4j

* Incorrect -

Spring Data

* Correct -

H2

* Your choice: incorrect -

Hibernate

* Incorrect -

I don't know yet.

29. Which Selenium method allows you to run JavaScript code in a browser?

* Your choice: correct -

executeScript

* Incorrect -

runScript

* Incorrect -

executeJS

* Incorrect -

runJS

* Incorrect -

I don't know yet.

30.What does the following configuration do?

<build>

<plugins>

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>findbugs-maven-plugin</artifactId>

<version>3.0.5</version>

<configuration>

<effort>Max</effort>

<threshold>Low</threshold>

</configuration>

<executions>

<execution>

<id>analyze-compile</id>

<phase>compile</phase>

<goals>

<goal>check</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

* Incorrect -

It enables the analysis level that optimizes memory, reports all bugs, and doesn't fail the compile process, if there are any.

* Incorrect -

It enables the analysis level that finds more bugs, reports only the critical ones, and doesn't fail the compile process, if there are any.

* Your choice: correct -

It enables the analysis level that finds more bugs, reports all bugs, and fails the compile process, if there are any.

* Incorrect -

It enables the analysis level that optimizes memory, reports only the critical ones, and fails the compile process, if there are any.

* Incorrect -

I don't know yet.

31.Which set of steps produces a characterization test for a legacy code base?

* Incorrect -

1. Select a new feature to implement;

2. Write a failing test for that feature;

3. Implement the feature;

4. Make the test pass.

* Incorrect -

1. Select a piece of code;

2. Refactor it to make it testable;

3. Write the test.

* Incorrect -

1. Select a new feature to implement;

2. Implement the functionality;

3. Write the test based on that functionality.

* Your choice: correct -

1. Select a piece of code;

2. In a test, write an assertion that you know will fail;

3. Let the failure show you the behavior of the code;

4. Change the test so that it expects the behavior that the code produces.

* Incorrect -

I don't know yet.

32.The Selenium Grid YAML file can contain various parameters. What is the remoteControlPollingIntervalInSeconds parameter meant for?

* Incorrect -

It represents how often the Hub will check for test execution results from clients.

* Correct -

It represents how often the Hub will check for registered Remote Controls status and idle testing sessions.

* Incorrect -

It represents how long a testing session can be idle before the Hub automatically unregisters the associated Remote Control.

* Your choice: incorrect -

It represents a general timeout value that is used by the Hub for all polling operations.

* Incorrect -

I don't know yet.

1. What does "Asserting first" mean?

---When creating a test, determine the expected result first.

2. Which option represents a source for parameterized tests in JUnit 5?

---A CSV (.csv) file

3. In the behavior-driven development (BDD) framework Cucumber, which annotations are used to match steps with step definitions?

---@Given, @When, @Then

4. In the following instance, what is the purpose of the @Primary annotation?

@Profile("test")

@Configuration

class TestConfiguration {

@Bean

@Primary

public UserManager userService() {

return Mockito.mock(UserManager.class);

}

}

---@Primary forces the Spring framework to return this instance when used in the 'test' profile.

5. What is a principle of test-driven development?

----Write a test and then write enough code so that the test passes.

6. Why is mutation testing considered a form of white-box testing?

----Because it involves changes to the source code.

7. What is wrong with the following JUnit5 test class?

public class AccountTest {

private Account account;

@BeforeAll

void init() {

this.account = new Account();

}

@Test

void testDeposit() {

this.account.deposit(1000);

assertEquals(1000, this.account.getBalance());

assertEquals(1, this.account.getNumberOfTransactions());

}

@Test

void testWithdrawal() {

this.account.withdrawal(200);

assertEquals(800, this.account.getBalance());

assertEquals(2, this.account.getNumberOfTransactions());

}

}

---The second test depends on the other. Unit tests should work independently.

8. In an ideal testing pyramid, which layer should contain the highest coverage and lowest coverage?

-----Unit testing - highestUser interface (UI) automated testing - lowest

9. How can you apply the following rule of Test-driven Development? Only write code to fix a failing test

---By writing a failing test first, and then writing the code that makes it pass.

10. Which option describes a good practice about multiple assertions in a test method?

Correct -It can have many physical assertions that form a single logical assertion about one action.

Your choice: incorrect -It can have many physical assertions about one action but only if they are the same (e.g. all the assertions are assertEquals).

Incorrect -It can have many physical assertions about many actions.

Incorrect -It can only have one physical assertion about many actions.

Incorrect -I don't know yet.

11. Select a feature/story which was written correctly in Gherkin:

Correct -Feature: Login As a user I must be able to login to the application to access my private stuff.

Incorrect -Feature(Login, As a user I must be able to login to the application to access my private stuff)

Incorrect -Story: Login

As a user I must be able to login to the application to access my private stuff.

Your choice: incorrect -Description: As a user I must be able to login to the application to access my private stuff

Feature: Login

Incorrect -I don't know yet.

12. Which of the following is a benefit of following the principles of test-driven development?

Your choice: incorrect -They make you write the test in an object-oriented way.

Incorrect -The application will have a high level of security.

Correct -The application code will be highly testable.

Incorrect -It builds a strong collaboration between stakeholders and developers.

Incorrect -I don't know yet.