1.

You are testing a function that sorts integers inside of an array using the Red-Green-Refactor

cycle. Once you are considered "in the green", what is the next step?

Your choice: correct -

Think about how you can optimize your codebase.

2.

In the following Gherkin feature file, which line describes the feature?

Feature: Is it Wednesday yet?

Everybody wants to know when it's Wednesday

Scenario: Monday isn't Wednesday

Given today is Monday

When I ask whether it's Wednesday yet

Then I should be told "No"

Correct:

Everybody wants to know when it's Wednesday

3.

Using Maven, which is an example of the required configuration for running tests with either the

tag "integration" or "feature-1"?

Correct:

...

<build>

<plugins>

...

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.19.1</version>

<configuration>

<properties>

<groups>integration | feature-1</groups>

</properties>

</configuration>

</plugin>

</plugins>

</build>

...

4.

You create the following code using Mockito to introduce dependency injection into your

application:

class A {

@Inject B b;

}

class B {

@Inject A a;

}

This leads to a circular dependency. How can you break this circular dependency?

Correct:

Inject a Provider<A> and Provider<B> instead of A and B.

5.

What action can you associate with behavior-driven development?

Correct:

Injecting the features

6.

You have a declared ChromeOptions:

ChromeOptions options = new ChromeOptions();

Which code snippet can you then use to correctly configure Chrome driver to accept all insecure

certs?

Correct:

options.setAcceptInsecureCerts(true);

options.setUnhandledPromptBehavior(UnexpectedAlertBehavior.ACCEPT);

WebDriver driver = new ChromeDriver(options);

7.

What is the difference between test-driven development (TDD) and traditional testing?

Correct:

With TDD, more focus is put on verifying whether testing will work properly. With traditional

testing, more focus is put on test case design.

8.

Which is a good practice when using test-driven development?

Correct:

Write new code only when a test fails.

9.

Given the following test:

public class TestPhone {

@Test

void testValidPhoneNumberLength() {

Phone phone = new Phone();

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

}

}

And the following class:

public class Phone {

public boolean validate(String phoneNumber) {

if(phoneNumber.replace("-", "").length() == 10) {

return true;

} else {

return false;

}

}

}

When using test-driven development, what would be the next step?

Correct:

Check if something needs to be refactored.

10.

You must build integration tests for an ecommerce system that customers primarily access

through a user interface. A portal also provides scaffolding for developers to test functionality. In

the backend, a service layer provides data to the client. And finally, a database holds state for

the application. Which layer is the least suited to integration testing?

Correct:

The developer portal

11.

The following method has a cyclomatic complexity of 3:

int myMethod(int n) {

if(n > 0) {

if(n < 100) {

return 1;

} else {

return 10;

}

}

return -1;

}

What is true about it?

Correct:

Because of its cyclomatic complexity, the method is easy to test.

12.

You are testing a drag and drop algorithm on a website you are developing. The algorithm

consists of mousedown, mousemove, and mouseup actions. How would you test the

mousedown phase?

Correct:

Create a clone of the element being moved.

13.

Which is a bug in the CORRECTNESS category of FindBugs?

Correct:

An apparent infinite loop (IL\_INFINITE\_LOOP)

14.

When using Spring and Mockito together, how can you ensure that the mocks initialized in a test

will be marked for closure after the method completes?

Correct:

By annotating each test with @DirtiesContext

15.

In which category does FindBugs classify the x%2 == 1 condition for finding the odd numbers?

Correct:

Dodgy code

16.

What is an example of a Hamcrest matcher?

Correct:

not

17.

What design principle is being applied to the following code?

public class MyTest {

private PersonRepository repo;

private PersonService service;

@BeforeEach

void init() {

repo = new FakePersonRepository();

service = new PersonService(repo);

}

@Test

void myTest() {

Person p = service.getPerson(1);

// ...

}

}

Correct:

The Liskov substitution principle

18.

Using PITest and Maven, what configuration do you need to activate a mutation such as the

following?

Original:

public int someNonVoidMethod() {

return 5;

}

public void foo() {

int i = someNonVoidMethod();

// ...

}

Mutated:

public int someNonVoidMethod() {

return 5;

}

public void foo() {

int i = 0;

// ...

}

Correct:

<configuration>

<mutators>

<mutator>NON\_VOID\_METHOD\_CALLS</mutator>

</mutators>

</configuration>

19.

Given the test class with line numbers to the left:

1) public class OrderTest {

2) @Test

3) void testDiscount() {

4) Order order = new Order();

5) order.setTotal(0.0);

6) assertEquals(0.0, order.calculateDiscount());

7) }

8) }

And the class:

1) public class Order {

2) private double total;

3) //...

4) public double calculateDiscount() {

5) double discount = 0.0;

6) if(this.total > 99.99) {

7) discount = this.total \* Discount.LEVEL2\_PERCENTAGE;

8) } else {

9) discount = this.total \* Discount.LEVEL1\_PERCENTAGE;

10) }

11) return discount;

12) }

13) }

What is the line coverage percentage of the method calculateDiscount?

Correct:

83% (5/6)

20.

What is true about Assert methods?

Correct:

Only failed assertions are recorded.

———————————————————————————————————————————

———————————————————————————————————————————

1. Using FindBugs/SpotBugs, what value can you set in the rank configuration to find

bugs with the rank 'SCARIEST'?

Correct:

3

2. Which assertion method comes with frameworks such as JUnit and TestNG?

Correct:

assertSame

3. 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 a 10% discount;

A purchase in the range of $50 to $99.99 has a 20% discount;

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

According to boundary-value analysis, which is an example of the minimum set of

values you should test?

Correct:

0.99

1

49.99

50

99.99

100

4. In FindBugs, which bug categories are usually left unchecked by default?

Correct:

Internationalization, Malicious code vulnerability, Security, Experimental

5. Which hierarchy arranges types of automation testing from highest to lowest test

complexity?

Correct:

Functional, Integration, Unit

6. Which dependency injection framework(s) supports Java Specification Requests

(JSR) 330?

Spring, Google Guice, and Dagger

7. Which package would you import to use the @Given annotation in Cucumber 6?

Correct:

io

8. Which tool can help you look for coding errors in Java code?

Correct:

SpotBugs

9. Which syntax for Cucumber tags executes tests tagged as FunctionalTests but

ignores tests tagged as IntegrationTests?

Correct:

tags={"@FunctionalTest","~@IntegrationTest"}

10. Which scenario is a good example of a behavior-driven development (BDD)?

Correct:

You are signing up for a Pluralsight social network;

You enter the required details and submit;

Then a Pluralsight social network account is created with your email;

And a confirmation email is sent to you.

11. Where would you put the @Parameters attribute in the following JUnit testing code,

assuming that int1 and int2 are integers?

<Option 1>

public static Collection<Object[]> populateData() {

Object[][] populateData = new Object[][] { {1,2},{3,4},{5,6} };

<Option 2>

return Arrays.asList(populateData);

}

<Option 3>

public void testMultiply() {

JUnitTestClass test = new JUnitTestClass();

assertEquals("Result", int1 \* int2, test.multiply(int1, int2));

}

<Option 4>

class JUnitTestClass {

public int multiply(int i, int j) {

return i \* j;

}

}

Correct:

Option 1

12. Given:

public class UserService {

private UserRepository userRepository;

public UserService(UserRepository userRepository) {

this.userRepository = userRepository;

}

public String renderNameId(long id) {

User user = userRepository.getUserById(id);

return formatNameId(user);

}

public String formatNameId(User user) {

// ...

}

}

Which code shows one way to test the method renderNameId in the Chicago test-driven

development (TDD) style?

Correct:

@Test

public void testRenderNameId() {

UserService userService = new UserService(new FakeUserRepository());

assertEquals("001. John Doe", userService.renderNameId(1));

}

13. You must scrape data from a website that has 500,000 pages. Your project is

backed up with good funding and requires the data collection work to be completed in

the least amount of time. What step can you take to extract data without being

recognized as a robot?

Correct:

Use proxies backed up by CAPTCHA's solving service to initiate requests in a non-sequential

pattern.

14. Based on the following String variable, which of these Hamcrest tests will pass?

String val = "Hello World";

Correct:

assertThat(val,

allOf(startsWith("Hello"), containsString("World")));

15. What is mutation testing?

Correct:

A type of testing where a statement is changed to see if the test cases are able to find an error

16. Which tools help solve some of the problems associated with Java Enterprise

Edition testing?

Correct:

Arquillian and ShrinkWrap

17. You created three files, features/Search.feature, runner/TestRunner.java,

and step/StepDefinitions.java, in separate sub-directories of the src/test folder. When

running the Search.feature file you receive an error stating No class found runner.Runner. What

fixes are required to run the file without any errors?

Correct:

Change the JUnit test class to runner.TestRunner.

18. In mutation testing, what problem do you create by

replacing == with < and y++ with ++y in the following method?

int myMethod() {

int a = 1;

int y = 1;

// Code that may change the values of a and y...

if(a == y++) {

return 1;

} else {

return 0;

}

}

Correct:

The mutations could cancel each other.

19. You are determining if two String variables are the same. Given the following code,

what is the result of this assertion?

@Test

public void testAsserts() {

String string1="test";

String string2="test";

assertSame(string1, string2);

}

Correct:

It returns true because both strings are "test".

20. Given the following JUnit 5 test class:

public class MyTest {

@DisplayName("My Test")

@ParameterizedTest(name = "{index} => {0}")

@ValueSource(strings = { "4", "5", "6" })

void myTest(int param) {

assertTrue(param > 0);

}

}

Which statement is true?

Correct:

When executing this test from the command line, part of the output will

be:

MyTest 1 = > 4 2 = > 5 3 = > 6

1. 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;

A sum of a negative number will throw a runtime exception;

Numbers bigger than one billion 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?

public class Calculator {

public int sum(int a, int b) {

return 3;

}

}

2. In Mockito 3, which command creates a stub that lets you verify interactions on a method?

mock()

5. What is a difference between using the mouseenter event and the mouseover event when handling user action?

The location of pointers inside a descendent element does not matter when the mouseenter event triggers, but it does matter when the mouseover event triggers.

9. In this scenario step, you have a book list that contains the following records:

| isbn | title | price |

| I2389423 | Cucumber for Java | 35.0 |

| I8459827 | Mastering Gherkin | 55.0 |

You have a class named book of the following structure:

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

public void validateBooks(List<Book> booksList) {

// test code

}

10. Which is something a static, non-configured code analysis tool can check regarding coding standards?

If a utility class (which contains only static methods or fields) does not have a public constructor

17. What is the goal of triangulation?

To prove that a piece of code is not general enough

18. What is the result of the following code?

ArrayList mock = mock(ArrayList.class);

mock.add(0, "Data");

System.out.println(mock.size());

"0" is printed as the output.

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 a 10% discount;
* A purchase in the range of $50 to $99.99 has a 20% discount;
* A purchase of $100 or more has 30% discount.

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

 Incorrect -

1

49.99

50

99.99

101

 Your choice: correct -

0.99

1

49.99

50

99.99

100

 Incorrect -

1

49.99

50

99.99

100

101

 Incorrect -

1

50

100

 Incorrect -

I don't know yet.

You have the following test methods:

* Two when() methods
* Two assertEquals() methods
* One verify() method

You run the test class in Java Mockito. What happens?

 Correct -

The test passes without using any additional method(s).

 Incorrect -

The test passes if you add verifyNoMoreInteractions(mock\_service);.

 Incorrect -

The test passes if you add verifyNoInteractions(mock\_service);.

 Your choice: incorrect -

The test passes if you add verify(mock\_service).method\_to\_be\_tested(7, 7);.

 Incorrect -

I don't know yet.

When working with testing in Java, what is test-driven development (TDD)?

 Correct -

A software development approach in which test cases are created to validate what the code will do

 Incorrect -

The act of combining independent pieces of software and determining if they work correctly when combined

 Your choice: incorrect -

A software development approach in which test cases are created to show the execution of the application to satisfy requirements

 Incorrect -

The act of testing existing code while simultaneously developing new code in a continuous pattern

 Incorrect -

I don't know yet.

You create the following code using Mockito to introduce dependency injection into your application:

class A {  
 @Inject B b;  
}  
  
class B {  
 @Inject A a;  
}

This leads to a circular dependency. How can you break this circular dependency?

 Incorrect -

Move the @Inject annotations above the class declarations.

 Correct -

Inject a Provider<A> and Provider<B> instead of A and B.

 Your choice: incorrect -

Inject a Provider<A> and Provider<B> instead of A and B and use the get() method.

 Incorrect -

Make the A and B class public instead of having no declared accessibility.

 Incorrect -

I don't know yet.

Given the following report from a mutation testing tool:

Total mutations: 10  
Mutations processed: 10  
Mutations not processed: 0  
Removed mutations: 7  
Survived mutations: 3  
  
Which is a valid conclusion after applying some basic analysis?

 Incorrect -

More unit tests or datasets are needed to decrease the number of processed mutations.

 Correct -

More unit tests or datasets are needed to increase the number of removed mutations.

 Your choice: incorrect -

Fewer unit tests or datasets are needed to increase the number of total mutations.

 Incorrect -

Fewer unit tests or datasets are needed to decrease the number of survived mutations.

 Incorrect -

I don't know yet.

Consider the following story:

Being a software engineer, I want my system to determine the CPU utilization limit and stop the new processes when the limit is reached without requiring my attention.

Which requirement describes an elaborated acceptance test after applying the behavior-driven development process to this story?

 Incorrect -

Given the CPU utilization is 80%  
When the CPU utilization increases  
Then the two new processes should be terminated

 Incorrect -

Given the CPU is being utilized  
When the CPU utilization increases  
Then new processes increase the CPU utilization but not above the limit

 Correct -

Given the CPU utilization is 80%  
When the CPU utilization changes to 65%  
Then the two new processes should be terminated within 1 second

 Incorrect -

Given the CPU is being utilized  
When the CPU utilization crosses a threshold  
Then new processes should be stopped

 Incorrect -

I don't know yet.

What is the most important aspect of testability?

 Incorrect -

Accessibility

 Incorrect -

Vulnerability

 Your choice: incorrect -

Controllability

 Correct -

Usability

 Incorrect -

I don't know yet.

Using Cucumber and JUnit with the following code snippet, how can you convert the DataTable to a list of lists correctly?

@When("^I enter valid data on the page$")  
public void enterData(DataTable table) {  
 //Initialize data table  
 //ANSWER HERE  
 System.out.println(data.get(1).get(1));  
}

 Incorrect -

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

 Correct -

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

 Your choice: incorrect -

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

 Incorrect -

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

 Incorrect -

I don't know yet.

Which Java code snippet best represents a fake?

 Incorrect -

class FooDouble implements FooInterface {  
 public String bar() {  
 return "baz";   
 }  
}

 Your choice: incorrect -

class FooImplDouble implements FooInterface {

}

 Incorrect -

class CannedEmptyUserDb implements DataBase {  
 public User get(String id) {  
 return null;  
 }  
   
 public Map<String, User> getAll() {  
 return new HashMap<>();  
 }  
   
 public User save(String id, User user) {  
 return null;  
 }  
   
 public User remove(String id) {  
 return null;  
 }  
}

 Correct -

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 -

I don't know yet.

You create a Cucumber project with the required files and run the feature file twice. It results in the following prompt:

First time:  
  
1 Scenarios (1 undefined)  
3 Steps (3 undefined)  
0m0.040s  
Missing implementation steps goes here ....  
  
  
Second time after adding the missing steps:  
  
1 Scenarios (1 pending)  
3 Steps (2 skipped, 1 pending)

The scenario is still in a pending state. What fixes are required to pass it?

 Incorrect -

Replace all throw new io.cucumber.java.PendingException(); statements with throw new io.cucumber.java.Execute(); statements.

 Incorrect -

Assign the name of the feature and step definition folders inside the runner class using the @CucumberOptions method.

 Incorrect -

Keep the name of all three methods similar to the string of their corresponding clause.

 Your choice: correct -

Replace all throw new io.cucumber.java.PendingException(); statements with System.out.println(); statements.

 Incorrect -

I don't know yet.

You have the following list:

List<String> mList = mock(MyList.class);

In the Mockito framework, which snippet accurately verifies that a mock object has never been used from a specific point in time?

 Your choice: correct -

verifyNoInteractions(mList)

 Incorrect -

verifyZeroInvocations(mList)

 Incorrect -

verifyAllInvocations(mList)

 Incorrect -

verifyNoInvocations(mList)

 Incorrect -

I don't know yet.

Using FindBugs/SpotBugs, what value can you set in the rank configuration to find bugs with the rank 'SCARIEST'?

 Incorrect -

10

 Correct -

3

 Incorrect -

20

 Incorrect -

7

 Incorrect -

I don't know yet.

Which test correctly configures the mock to throw a CantAddNullsException when inserting a null, assuming that you are using the Mockito framework?

 Incorrect -

@Test(expected = CantAddNullsException.class)  
 public void throwTest() {  
 ArrayList mock = mock(ArrayList.class);  
 when(mock).add(isA(Integer.class), isNull()).doThrow(new   
 CantAddNullsException());  
 mock.add(0, null);  
}

 Your choice: correct -

@Test(expected = CantAddNullsException.class)  
 public void throwTest() {  
 ArrayList mock = mock(ArrayList.class);  
 doThrow(new   
 CantAddNullsException()).when(mock).add(isA(Integer.class),   
 isNull());  
 mock.add(0, null);  
}

 Incorrect -

@Test(expected = CantAddNullsException.class)  
 public void throwTest() {  
 ArrayList mock = mock(ArrayList.class);  
 doThrow(new   
 CantAddNullsException()).when(mock).add(isA(Integer.class),   
 isNotNull());  
 mock.add(0, null);  
}

 Incorrect -

@Test(expected = CantAddNullsException.class)  
 public void throwTest() {  
 ArrayList mock = mock(ArrayList.class);  
 when(mock).add(isA(Integer.class),   
 isNull()).thenThrowException(new CantAddNullsException());  
 mock.add(0, null);  
}

 Incorrect -

I don't know yet.