NAME:

Question 1.

The cyclomatic complexity of the code influences how easily it can be read and maintained. What factors can influence the complexity metric, as defined in SonarQube?

Select one:

A. Number of variables declared in a method.

B. Number of iterations performed in cycles (repetitions of a cycle).

C. Size of a class (number of lines).

D. Number of variables declared in a method that are not used.

E. Occurrence of statements that divide the flow control, such as `if, for, while, switch.`.

Question 2.

TDD is use in agile methods as a strategy to ensure that a relevant test base is available at each iteration.

What is a distinctive characteristic of TDD?

Select one:

A. All tests must always be passing (prevents code regressions).

B. No more tests should be written than is strictly necessary to exercise the existing code (otherwise tests would be superfluous).

C. Test plans shall be written in the Analysis phase when developing product specifications before all development.

D. The daily routine of the developer is: clean/improve the previous code; add a new test; implement the code needed for the new test to pass.

E. Increments in production code must be preceded by one or more tests that make that code necessary.

Question 3.

Consider the following description of a test included in a Spring Boot application: "This test seeks to confirm that the JSON objects in the response are being mapped correctly when a valid object is used in the request, and when the requests are not valid, that we are sending the correct HTTP error along with a descriptive reason for the failure." [In: report of a TQS project]

What type of test (in Spring Boot) is most efficient/suitable for the stated purpose?

Select one:

A. @SpringBootTest

B. @WebMvcTest

C. @ExtendWith(MockitoExtension.class)

D. @MockMvc

E. @SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)

Question 4.

The excerpt shows the contents of the application-db.properties file of a SpringBoot project.

What is its usefulness in the context of integration testing?

```
spring.datasource.url=jdbc:mysql://localhost:3306/demo
spring.jpa.hibernate.ddl-auto=create-drop
spring.datasource.username=demo
spring.datasource.password=demo
```

Select one:

A. Is used to create a persistent data context over the various test runs so that each CI cycle can use the data (in the database) from the previous cycle.

B. Is used to define a temporary database, for example, configured in the annotation @TestPropertySource

C. Is used to set up a temporary, in-memory test database that is destroyed at each test run.

D. It is not used in tests; it would have to be called "application.properties" to be used by the annotation @AutoConfigureTestDatabase.

E. It is not used in tests; it is used to configure the *runtime* (production) database.

Question 5.

In the context of unit test that use the Mockito framework, when should the programmer @Spy an object?

A. The @Spy annotation is not useful for unit tests; it should be used to cache the response of remote dependencies (in integration tests).

B. @Spy and @Mock can be use interchangeably to generate a “dummy” test object, without implementation.

C. @Mock objects define their behavior through expectations; @Spy objects automatically infer the expected behavior of the mock object, by learning from previous invocations.

D. @Spy allows to use the real implementation of certain methods of an object, while defining expectations for other methods.

E. @Spy is used in Spring Boot tests to include detailed feedback in the server log, for test debugging.

Question 6.

The BDD approach stresses the need for testing in a QA strategy.

Which of the following features is **FALSE** with respect to BDD practices?

Select one:

A. New scenarios become easier to implement as more "steps" are implemented, since common steps can be shared/reused.

B. The description of features can be seen "executable specifications" because they directly provide control logic and data for the testing code.

C. It does not require the use of other requirements specification techniques, since it allows to centralize the product specification in the description of the "feature", which is subject to version control with the rest of the code.

D. It is supported by a natural language that allows you to describe the test in the terms of the problem domain, understood by all team.

E. Facilitates understanding of the expected behavior of the software by fostering collaboration between the technical team and the "business" team.

Question 7.

The code excerpt shows a common testing scenario using standard Spring Boot practices.

What is this code testing for (i.e., what can one conclude when the test passes)?

A. The GreetingService contains a method “greet()” that return the concatenation of “Hello” and the argument passed in the request.

B. The GreetingService uses a mocked instance of the GreetingController that answers HTTP requests in the endpoint “/greeting”.

C. There is an endpoint “/greeting” available for invocation, provided by GreetingController, without mandatory arguments, that returns a plain text response.

D. The GreetingService should return a JSON response which always include “Hello, Mock”.

E. GreetingController should define a mapping for “/greeting” which is to be invoked with a String argument containing “Hello, Mock.”.

```java
@WebMvcTest(GreetingController.class)
public class WebMockTest {

        @Autowired
        private MockMvc mockMvc;

        @MockBean
        private GreetingService service;

        @Test
        public void greetingShouldReturnMessageFromService() throws Exception {
                when(service.greet()).thenReturn("Hello, Mock");

this.mockMvc.perform(get("/greeting")).andDo(print()).andExpect(status().isOk())

.andExpect(content().string(containsString("Hello, Mock")));
        }
}
```