Practical exercises and questions:

In the following section there will be six exercises about Cross Site Scripting and SQL injection. We suggest you to follow the exercises in the order in which they are presented.

Exercise 1: In the following exercise you will experience what happens if countermeasures can be easily bypassed.
You can find the lab here: https://portswigger.net/web-security/cross-site-scripting/contexts/lab-attribute-angle-brackets-html-encoded

Objective: The goal of the exercise is to perform a Cross Site Scripting attack that injects an attribute and calls the alert function. The only thing you know is that this lab contains a Cross Site Scripting vulnerability in the search functionality where angle brackets are HTML encoded.

A suggestion in case you are lost: *Try to understand how a legitimate request works. How does the response page handle input data? You might be able to manipulate the results to suit your needs.*

Exercise 2: In the following exercise you will experience what happens if users can manipulate the content of the page.
You can find the lab here https://portswigger.net/web-security/cross-site-scripting/dom-based/lab-document-write-sink-inside-select-element

Objective: The goal of the exercise is to perform a Cross Site Scripting attack that breaks the select element and calls the alert function. The only thing you know is that this lab contains a Cross Site Scripting vulnerability in the stock checker functionality. It uses the document.write function which writes data out to the page. This function is called with data from location.search which you can control using the website URL. The data is enclosed in a select element.

A suggestion in case you are lost: *Look at the page source and try to understand if you can manipulate some parameters. You may find something interesting to exploit.*

Exercise 3: In the following exercise you will experience what happens if an application uses vulnerable functions to change the content of the page.
You can find the lab here: https://portswigger.net/web-security/cross-site-scripting/dom-based/lab-innerhtml-sink

Objective: The goal of the exercise is to perform a Cross Site Scripting attack that calls the alert function. The only thing you know is that this lab contains a Cross Site Scripting vulnerability in the search functionality. It uses an innerHTML assignment, which changes the HTML contents of a div element, using data from location.search.

A suggestion in case you are lost: *Try to understand how the search results page is populated, you may find something interesting to exploit. Sometimes you can search strange things, what happens if you search for something "wrong"?*

Exercise 4: In the following exercise you will experience what happens if the results returned by the query are returned in the application response and can be easily manipulated.
You can find the lab here: https://portswigger.net/web-security/sql-injection/union-attacks/lab-determine-number-of-columns

Objective: The goal of the exercise is to determine the number of columns returned by the query by performing a SQL injection UNION attack that returns and additional row containing null values. The only thing you know is that the lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables.

A suggestion in case you are lost: *Try to understand how many columns are returned by the query. Null values and errors returned by the application can be very useful.*

Exercise 5: The following exercise is another step of the previous attack. You can determine which columns return string data.
You can find the lab here https://portswigger.net/web-security/sql-injection/union-attacks/lab-find-column-containing-text

Objective: The goal of the exercise is to perform a SQL injection UNION attack that returns an additional row with a specific value provided by the application. This technique is used to determin which columns are compatibile with string data. The only thing you known is that the lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables.

A suggestion in case you are lost: *The lab will provide you with a value, use that value to your advantage. Null values and errors returned by the application can be very useful again.*
Exercise 6: Use the techniques learned in the previous labs to attack this application, you might be able to log yourself in as an administrator!
You can find the lab here: https://portswigger.net/web-security/sql-injection/union-attacks/lab-retrieve-data-from-other-tables

Objective: The goal of the exercise is to perform a SQL injection UNION attack that returns an additional row with a specific value provided by the application. This technique is used to determin which columns are compatibile with string data. The only thing you known is that the lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables.

A suggestion in case you are lost: *Using the discovered information you might be able to make the application return the entire table containing username and passwords of the application users.*

Questions:

In this little section you will find some simple questions to test your comprehension of the topic.

1. Briefly describe the different types of XSS attack.

2. Briefly explain when an application can be vulnerable to injection type attacks.

3. Which aspects of data can be affected by SQL injection attacks?