## Error Handling in Java (including functional programming):

1. **Question:** What is the difference between checked and unchecked exceptions in Java ?

**Answer: C.**

1. Checked exceptions are those that occur during compilation, while unchecked exceptions occur at runtime.
2. Checked exceptions should always be caught with a try/catch block.
3. Checked exceptions (inherit from Exception) must be handled explicitly in the code or declared in the method signature. Unchecked exceptions (inherit from RuntimeException) do not require explicit handling.
4. Unchecked exceptions are more serious than verified ones.
5. **Question: What is** try / catch block and how is it used to handle exceptions?

**Answer:** B.​

1. The try/catch block is a control structure used to handle errors at compile time.
2. The try block contains code prone to generating exceptions. If an exception occurs, the catch block catches and handles the error.
3. The try/catch block is used to define local variables in Java.
4. The try/catch block is optional and does not affect the normal flow of the program.
5. **Question:** What is exception propagation in Java?

**Answer:** A.

1. Exception propagation occurs when an exception is not handled in a method and propagates up the call stack.
2. Exception propagation refers to the process of creating new exceptions at runtime.
3. Exception propagation occurs in static methods, when a stack call is made.
4. Exception propagation occurs in checked exceptions and what it does is call the error stack and introduce a new error.
5. **Question:** Which Spring annotation is used to handle exceptions globally throughout the application?

**Answer: A.**

1. @ControllerAdvice
2. @ExceptionHandler
3. @HandleError
4. @ErrorHandle
5. **Question:** How are errors handled in functional programming?

**Answer:** C.​

1. Errors in functional programming are not handled as they are considered part of the normal flow of the program.
2. In functional programming, try/catch blocks are used just like in imperative programming.
3. In functional programming, types like Either or Try are used to explicitly represent success or error values.
4. Errors in functional programming are handled automatically without the need for programmer intervention.

## Functional Programming Paradigm in Java:

1. **Question:** What is the difference between errors and exceptions in Java?

**Answer:** C.​

1. Errors are exceptions that occur during compilation, while exceptions occur at runtime.
2. Errors are always checked, while exceptions can be checked or unchecked.
3. Errors are serious problems that cannot be recovered, while exceptions are used to handle recoverable failures within a program.
4. Errors are handled automatically by the system, while exceptions require explicit handling.
5. **Question:** What is the functional programming paradigm in Java?

**Answer:** B.​

1. An approach that relies on the sequence of statements to change the state of the program.
2. A programming style that treats functions as first-class objects and avoids changing state and mutable data.
3. An approach that focuses on object orientation and inheritance.
4. A method for handling errors at compile time.
5. **Question:** What is the difference between object-oriented programming and functional programming?

**Answer:** D.

1. Object-oriented programming uses only functions, while functional programming is based on objects.
2. Functional programming is more suitable for web applications, while OOP is better for desktop applications.
3. Functional programming does not allow the use of classes and objects.
4. OOP focuses on mutable objects and states, while functional programming is based on immutable functions and data.
5. **Question:** What is immutability in functional programming?

**Answer:** D.

1. Immutability refers to the ability of objects to change their state at runtime.
2. Immutability only applies to methods in functional programming.
3. Immutability only applies to primitive data types.
4. Immutability implies that objects cannot change once they are created. In Java, it is achieved by declaring fields as final.
5. **Question:** What is the concept of pure functions in functional programming?

**Answer:** B.

1. Pure functions are those that can modify the global state of the program.
2. Pure functions have no side effects and always produce the same result for the same arguments.
3. Pure functions are only used in pure functional programming languages.
4. Pure functions cannot accept arguments.