Congratulations! You passed!

Grade received 100% To pass 80% or higher

Go to next item

Module Review

Latest Submission Grade 100%

1. Which of the following are examples of **runtime exceptions**? (select all that apply) 1 / 1 point

- [x] Application error
  - Correct
    Yes. This is a runtime exception.
- [ ] Mathematical calculation error
- [x] Input error
  - Correct
    Yes. This is a runtime exception.
- [x] Divide by zero error
  - Correct
    Yes. This is a runtime exception.

2. What happens whenever you invoke a method that could throw a checked exception? 1 / 1 point

- ( ) The compiler will log a standard exception.
- ( ) The compiler will comment your code where the exception occurrs.
- (•) The compiler will insist that you handle the exception.
  - Correct

3. Which of the following are **common exceptions**? (select all that apply) 1 / 1 point

- [x] ArrayIndexOutOfBounds
  - Correct
    Yes. This is a common exception that you may incur.
- [x] SQLException
  - Correct
    Yes. This is a common exception that you may incur.
- [x] OutOfMemoryError
  - Correct
    Yes. This is a common exception that you may incur.
- [x] IOException
  - Correct
    Yes. This is a common exception that you may incur.
- [x] NullPointerException
  - Correct
    Yes. This is a common exception that you may incur.

4. In the following code example, what will happen if set.Speed throws a SpeedException? **1 / 1 point**

```
try {
    myCar.setSpeed(220);
} catch (SpeedException e) {
    System.out.println("Car is going too fast!");
}
```

- ○ Java will continue processing the current code block but will stop with an exception after the block.
- ○ Java will log the exception in the log file and will continue to execute the rest of the code.
- ◉ Java will skip any remaining code in the try block and execute the code in the catch block instead.

✓ **Correct**
Correct.

5. What is a **finally** statement used for? **1 / 1 point**

- ○ Identifying the last catch block in a try/catch statement.
- ○ Processing a single try statement without any catch blocks.
- ◉ Recovering resources and/or cleaning up following the execution of a set of statements.

✓ **Correct**

6. How will the exception will be processed based on the code below? **1 / 1 point**

```
try {
      myCar.setSpeed(1000);
}
catch(Exception e){/* notice that there's no code here */ }
```

- ○ The exception will be logged.
- ◉ The exception will be suppressed.
- ○ The exception will be set to 1000.
- ○ The exception will prompt you for a description.

✓ **Correct**
Correct.

7. Which of the following are examples of **exceptions**? (select all that apply) **1 / 1 point**

- ☑ An integer is divided by zero.

✓ **Correct**
Correct.

- ☑ An application tries to open a file that does not exist.

✓ **Correct**
Correct.

- ☐ A new network connection has been defined
- ☑ A program tries to access a record beyond the bounds of an array.

✓ **Correct**
Correct.

8. Where is the best place to deal with exceptions? **1 / 1 point**

- ◉ in the method where they could be thrown.

- ○ at the beginning of the next code block.
- ○ in the area where you define your variables.

Correct
Correct!

9. Which of the following are true about **checked exceptions**? (select all that apply) **1 / 1 point**

- ☑ Checked exceptions are the ones you want to force your client code to address.

Correct
Correct.

- ☐ Checked exceptions always follow a finally block.
- ☐ Checked exceptions are automatically checked and addressed, you don't have to code for them.
- ☑ Checked exceptions have to be declared in throws clauses.

Correct
Correct.

10. True or False? As a shortcut, you can use exceptions in your normal control flow. **1 / 1 point**

- ○ True
- ◉ False

Correct
Correct. Use exceptions only in exceptional conditions, not as a result of normal control flow.