**DEMOGRAPHIC INFORMATION**

1.- Gender: Male ( ) Female ( )

2.- Age: \_\_\_\_\_\_\_\_\_

3.- Studies: PhD ( ) Master ( ) Bachelor ( ) Student ( ) None ( )

4.- Years of experience in software development: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5.- Years of experience in Java software development: \_\_\_\_\_\_\_\_\_\_\_\_

6.- Rate from 1 (None) to 5 (Expert) your level of knowledge of Java: 1( ) 2( ) 3( ) 4( ) 5( )

**EVALUATION**

**A.** The following Java methods are part of the "compiler" project, which is a compiler for a simple programming language, for use in education.

1. The method is included in the "SymbolTable" class, and has the following code:

**public** **boolean** \_\_\_\_\_\_\_\_\_\_\_\_\_(String lexeme) {

**for** (Symbol symbol : symbols) {

**if** (symbol.lexeme.equals(lexeme)) {

functionType = symbol.kind.toString();

returnType = symbol.returnType;

numOfVar = symbol.numOfVar;

**return** **true**;

}

}

**return** **false**;

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| symbolTableMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| findSubroutineSymbol | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| findLexeme | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| findFunctionSymbol | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

2. The method is included in the "Parser" class, and has the following code:

**private** **void** \_\_\_\_\_\_\_\_\_\_\_\_\_() {

**try** {

vmFile = **new** File(lexer.getFolderPath() + File.***separator*** + className + ".vm");

**if** (!vmFile.exists()) {

vmFile.createNewFile();

}

} **catch** (Exception e) {

e.printStackTrace();

}

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| createVMFileIfNotExists | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| vmFileCreate | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| parserMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| createVMFile | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

**B.** The following Java methods are part of the "jvector" project, which is a pure Java embedded vector search engine, used by DataStax Astra DB and Apache Cassandra.

1. The method is included in the internal class “CachedView" of the class “CachingGraphIndex", and has the following code:

**public** **float**[] \_\_\_\_\_\_\_\_\_\_\_\_\_(**int** node) {

**var** cached = cache.getNode(node);

**if** (cached != **null**) {

**return** cached.vector;

}

**return** view.getVector(node);

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| getCachedOrViewVector | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| cachedViewMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| getVector | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

2. The method is included in the "GraphIndexBuilder" class, and has the following code:

**static** <T> **float** \_\_\_\_\_\_\_\_\_\_\_\_\_(

VectorEncoding encoding, VectorSimilarityFunction similarityFunction, T v1, T v2) {

**switch** (encoding) {

**case** ***BYTE***:

**return** similarityFunction.compare((**byte**[]) v1, (**byte**[]) v2);

**case** ***FLOAT32***:

**return** similarityFunction.compare((**float**[]) v1, (**float**[]) v2);

**default**:

**throw** **new** IllegalArgumentException();

}

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| scoreBetween | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| calculateSimilarityScore | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| graphIndexBuilderMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| calculateSimilarity | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

**C.** The following Java methods are part of the "Log4-detector" project, which is a scanner that detects Log4J vulnerabilities.

1. The method is included in the "VersionComparator" class, and has the following code:

**private** **static** **int** \_\_\_\_\_\_\_\_\_\_\_\_\_(String[] words, **boolean** isLastWord) {

**int** min = Integer.***MAX\_VALUE***;

**for** (**int** i = 0; i < words.length; i++) {

String word = words[i];

**boolean** isLastOfLast = isLastWord && (i == words.length - 2);

min = Math.*min*(min, *calculateScore*(word, isLastOfLast));

}

**return** min;

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| findMinScore | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| versionComparatorMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| findMinWordScore | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| minWordScore | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

2. The method is included in the "Strings" class, and has the following code:

**public** **static** String \_\_\_\_\_\_\_\_\_\_\_\_\_(String s) {

**if** (s == **null**) {

**return** **null**;

} **else** {

StringBuilder buf = **new** StringBuilder(s);

**for** (**int** i = s.length() - 1; i >= 0; i--) {

**char** c = s.charAt(i);

**if** (*isLetterOrDigit*(c)) {

**return** buf.toString();

} **else** {

buf.deleteCharAt(i);

}

}

}

**return** "";

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| stringMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| removeNonAlphaNumericCharacters | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| stripTrailingNonAlphaNumerics | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| removeTrailingNonAlphaNumerics | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

**D.** The following Java methods are part of the "ramen" project, which implements a simple social network, and was used in a Java programming course.

1. The method is included in the "Installer" class, and has the following code:

**private** **void** \_\_\_\_\_\_\_\_\_\_\_\_\_() {

**boolean** enableNext = **true**;

enableNext &= (path != **null**);

**for** (String s : files) {

**if** (s != **null**) {

File f = **new** File(s);

enableNext &= f.exists();

} **else**

enableNext = **false**;

}

next.setEnabled(enableNext);

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| updateNextButtonEnabledState | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| updateNext | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| installerMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| enableNextButton | 1( ) 2( ) 3( ) 4( ) 5( ) |  |

2. The method is included in the "Ramen" class, and has the following code:

**public** List<Answer> \_\_\_\_\_\_\_\_\_\_\_\_\_(Question q) **throws** ForbiddenAction {

**if**(currentUser.equals(q.getAuthor()))

**return** q.reviewAnswers();

**else**

**throw** **new** ForbiddenAction("Only question author can review its answers.");

}

For the following method names, indicate whether they are appropriate for the previous method. Rate from 1 (disagree) to 5 (strongly agree) your level of agreement.

|  |  |  |
| --- | --- | --- |
| **Method name** | **Is the name appropriate?** | **Comments (optional)** |
| ramenMethod | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| getAnswersForReview | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| getReviewedAnswers | 1( ) 2( ) 3( ) 4( ) 5( ) |  |
| reviewQuestion | 1( ) 2( ) 3( ) 4( ) 5( ) |  |