U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

M.EIC045 and MESW0004
Software Testing, Verification and Validation
2nd Exam

February 02, 2024

Student name: _____
Student number: _____

- **M.EIC**
- **MESW**

This *exam* is composed by two main groups:

- Group I is worth 25% (i.e., 5 points out of 20) and it is composed of 10 multiple choice questions.

(a) Note: There is only one correct answer per question!

(b) Note: Each correct answer is worth +0.5 points (out of 20) and each incorrect answer has a -0.25 penalty. In case you get a negative overall score in this Group, your score will be truncated to 0.

- Group II is worth 75% (i.e., 15 points out of 20) and it is composed of two open questions each with one or more sub questions.

Few additional details:

- You have 90 minutes to complete this exam.
  - In the first 30 minutes, you are not allowed to leave the room.
  - Once the first 30 minutes have passed, you are allowed to hand over your exam and leave the room.
- You should answer every single question to the best of your knowledge and you must write every single answer in the provided pieces of paper. No other pieces of paper will be considered/evaluated.
  - Group I must be answered in this document.
  - Group II must be answered in the provided/separated sheet.
- You are NOT allowed to use or consult any additional material (books, slides, etc) during the exam.
- The use of mobile phones or any other electronic devices is strictly prohibited during the exam and must be turned off.
- You may answer the open questions in Portuguese or English.

GROUP I (5 points)

1. Verification can be termed as “Are we building the product right?”.

- ☐ True.
- ☐ False.

2. Which of the following is a **FALSE** statement about software testing?

- ☐ Dynamic testing may find problems that static testing cannot.
- ☐ Dynamic testing detects failures.
- ☐ Static testing is worse than dynamic testing in detecting unused variables.
- ☐ Static testing should start before dynamic testing.

3. Which of the following is a **TRUE** statement about software testing?

- ☐ The primary goal of software testing is to exercise the software under test.
- ☐ One of the software testing goals is to ensure 100% correctness.
- ☐ Usually, exhaustive testing is unfeasible.
- ☐ Random testing is the most effective technique to find failures.

4. Which of the following is a **FALSE** statement about test planning?

- ☐ Test planning includes assigning resources to different test activities.
- ☐ Test entry criteria allow to define when to stop testing.
- ☐ Test planning estimates test effort based on metrics or based on experts.
- ☐ Examples of exit criteria are residual risks and code coverage.

5. Which of the following is a **FALSE** statement about test independence?

- ☐ The highest level of test independence occurs when developers test their own code.
- ☐ Outsourcing testing achieves the highest level of test independence.
- ☐ Test independence can lead developers to neglect quality responsibility.
- ☐ The benefit of test independence is to see other and different defects.

6. Which of the following is a **TRUE** statement about static testing?

- ☐ Technical reviews are performed by peers and vary from quite informal to very formal.
- ☐ Walkthroughs include a formal process based on rules / checklists with entry and exit criteria.
- ☐ An inspection is open-ended and optionally with pre-meeting preparation.
- ☐ An informal review is led by the author and the results achieved are always documented.

7. Which of the following is a **FALSE** statement about integration testing?

- ☐ One of the problems with integration testing is localising the errors.
- ☐ Examples of interfaces are shared memory and exchanged messages.
- ☐ Integration testing is black-box and usually handled by an independent team.
- ☐ Bottom-up integration testing requires the development of stubs.

8. According to requirement R1, the variable `A` can belong to [0-14[, [14-30[and [30-35]. Regarding that the graphical user interface does not allow inserting negative values, which of the following test cases achieve 100% coverage according to boundary value analysis for variable `A`?

- ☐ 0, 1, 14, 15, 16, 30, 31, 32, 34, 35.
- ☐ 0, 1, 14, 15, 16, 30, 31, 32, 34, 35, 36.
- ☐ 0, 1, 13, 14, 15, 29, 30, 31, 34, 35.
- ☐ 0, 1, 13, 14, 15, 29, 30, 31, 34, 35, 36.

9. According to the requirements, you get 3 equivalent classes for variable `A`, 3 equivalent classes for variable `B`, and 3 equivalent classes for variable `C`. What is the number of test cases that you get when combining the input values according to pairwise testing?

- ☐ 6 test cases.
- ☐ 9 test cases.
- ☐ 12 test cases.
- ☐ 27 test cases.

10. Consider the following program:

```
read(A); read(B); read(C);
if (C) {
    print "C";
    if (A and B)
        print "A and B";
} else {
    print "Not C";
}
```

What is the minimum number of test cases that you need to cover 100% statement coverage and 100% decision coverage?

- ☐ 2 test cases.
- ☐ 3 test cases.
- ☐ 4 test cases.
- ☐ 5 test cases.

GROUP II (15 points)

1. Data flow testing (9 points)

Consider the following function named `sameEnds` which returns the longest substring that appears at both the beginning and end of the string without overlapping.
For example, `sameEnds("abXab")` returns `ab`.

```java
   public String sameEnds(String string) {
1.     int length = string.length();
2.     int half = length / 2;
3.
4.     String left = "";
5.     String right = "";
6.
7.     int size = 0;
8.     int i = 0;
9.     while (i < half) {
10         left = left + string.charAt(i);
11.        right = string.charAt(length - 1 - i) + right;
12.
13.        if (left.equals(right)) {
14.            size = left.length();
15.        }
16.
17.        i++;
18.    }
19.
20.    return string.substring(0, size);
   }
```

1.1 Draw the control-flow graph (CFG) of the source code above. (+2 out of 9 points)

1.2 Create a *def-use* table of the `length`, `left`, and `size` variables. (You may ignore the parameter `string`.) Justify your answer. (+4 out of 9 points)
Tip: consider a maximum of two interactions of the loop.

1.3 Define a minimal set of test cases (i.e., inputs) that exercise all *def-clear* paths of the `left` variable. Justify your answer. (+3 out of 9 points)
Tip: consider a maximum of two interactions of the loop.

2. Structural coverage and Mutation testing (6 points)

Given the following function

```java
   public long mdc(long m, long n) {
1.     long d;
2.     if (n == 0) {
3.         return m;
4.     }
5.     if (m == 0) {
6.         return n;
7.     }
8.     d = Math.min(m, n);
9.     while (m%d != 0 || n%d != 0) {
10.        d--;
11.    }
12.    return d;
   }
```

2.1 Define a minimal set of test cases (i.e., inputs) that cover/exercise all conditions and all decisions. Justify your answer. (+3 out of 6 points)

2.2 Define a minimal set of test cases (i.e., inputs) that are able to *kill* all mutants described in the following table. In case your set of test cases is not able to *kill* all mutants, please justify why. Note: you may re-use some of the tests defined in 2.1. (+3 out of 6 points)

Id	Line number	Original code	Mutation
1	2	`if (n == 0)`	`if (n != 0)`
2	5	`if (m == 0)`	`if (m != 0)`
3	9	`while (m%d != 0 || n%d != 0)`	`while (m%d == 0 || n%d != 0)`