Written Examination
DIT636 / DAT560
Software Quality and Testing
March 19, 2025; 8:30 – 12:30

Course Responsible/Examiner: Gregory Gay
E-Mail: ggay@chalmers.se
Phone: +46 73 856 77 93
Examination Hall Visits: 9:30, 11:30
Allowed aids: No notes or other aids are allowed.
Grading Scale: 0-49 (U), 50-69 (3), 70 - 85 (4), 86-100 (5)
Examination Review: April 14, 9:00

There are a total of 9 questions and 100 points available on the test. On all essay type questions, you will receive points based on the quality of the answer - not the quantity. Illegible answers will not be graded.

Question 1 (Warm Up) - 10 Points

Multiple solutions may apply. Select all that are applicable. True/False worth 1 point each, multiple choice worth 2 points each.

1. For the expression `(a || !c) || (a && b)`, the test suite **(a, b, c) = {(T, F, T), (F, T, T), (T, T, T), (F, F, F)}** provides:
    a. MC/DC Coverage
    b. Decision Coverage
    c. Basic Condition Coverage
    d. Compound Condition Coverage

2. During exploratory testing, the couch potato tour recommends running the system for a long period of time (e.g., overnight) to see if there are failures that emerge over time.
    a. True
    b. False

3. A liveness property is a property that should hold over a path of unknown length.
    a. True
    b. False

4. Mutation is considered useful because a sequence of small code changes can approximately model a larger fault in the program.
    a. True
    b. False

5. You have designed the software for an ATM to meet the following requirement: “If the amount of remaining money in the machine cannot be calculated, then display an error screen and prevent users from performing any additional interactions until the functionality is restored.” Which type of property is this?
    a. Correctness
    b. Reliability
    c. Robustness

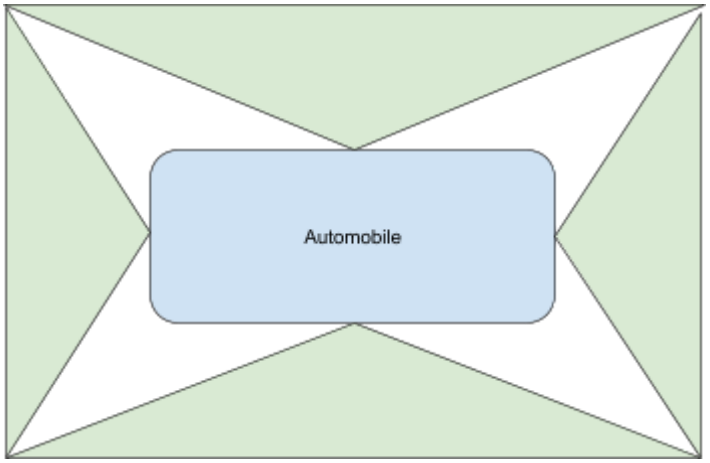
6. An invalid mutant is one that is detected by many test cases.
    a. True
    b. False

7. You are designing a pedestrian detection system for a vehicle. Which **one** of the following quality attributes would be of most importance to you?
    a. Reliability
    b. Performance
    c. Scalability

    Briefly (1-2 sentences), explain why this is the most important.

Question 2 (Quality Scenarios) - 10 Points

Consider a camera-based object detection system in an automobile. This system uses a set of four cameras to capture the area around a vehicle (the green shaded areas in the image below):



This system offers the following functionality:

- Detects lanes on the road, warning the driver if the car crosses any lines that should not be crossed.
- Detects objects in the roadway (e.g., pedestrians or obstacles) that should not be collided with, and warns the driver if they are in the path of the vehicle.
- Warns the driver if there are conditions where the system cannot make accurate predictions. For example, if weather conditions cause a sharp reduction in image quality or if the camera has been covered.

Create one **reliability** and one **availability** scenario for this system, with a Description, System State, System Environment, External Stimulus, Required System Response, and Response Measure for each.

Question 3 (Testing Concepts) - 8 Points

- Define system (integration) testing and exploratory testing.
- Explain how systems are tested at each stage.
- Explain how the two stages differ.
- Explain types of faults that may be more likely to be exposed by each of the two stages.

Question 4 (Test Design) - 12 Points

Recall the object discussion system discussed in Question 2.

Internally, the following method is invoked each time that the sensor data is refreshed:

```
public String[] analyzeObstacles (List<Float> cameraData)
```

The input parameter `cameraData` represents a list of zero or more potential obstacles detected by that camera. Each Float represents the distance that a detected obstacle is from the camera.

For example, if:
`System.out.println(cameraData.get(0));`
prints “1.45” to the screen, this indicates that the camera has detected an object (`cameraData.get(0)`) that is 1.45 meters from that camera.

This method should analyze the camera data and return a String array, where each item is a warning for the driver. The warnings should include:

- Any detected obstacle that is within 2 meters of the camera.
- Any situation where an obstruction may be blocking the camera (indicated by a detected object that is listed as 0.00 meters from the camera).

Perform functional test design for this method.

1. Identify choices (controllable aspects that can be varied when testing)
2. For each choice, identify representative input values.
3. For each value, apply constraints (IF, ERROR, SINGLE) if they make sense.

You do not need to create test specifications or concrete test cases. For invalid input, **do not** just write “invalid” - be specific. If you wish to make any additional assumptions about the functionality of this method, state them in your answer.

Question 5 (Exploratory Testing) - 8 Points

Exploratory testing typically is guided by “tours”. Each tour describes a different way of thinking about the system-under-test and prescribes how the tester should act when they explore the functionality of the system.

1. Describe one of the tours that we discussed in class **other than the supermodel tour**.
2. Consider a web-based discussion forum. This forum offers the following functionality:
   - Users can register for an account, using their e-mail address. They can choose a display name (must be unique) and a password for their account.
   - Once registered, users can log into their account.
   - Some users, called “administrators”, have additional privileges.
   - Administrators can create, edit the name and description of, and delete “boards”.
   - A “board” is an area where users can post topics related to the subject of that board.
   - Users can also reply to topics.
   - Users can edit their own posts in topics.
   - Administrators can also edit or delete posts in topics.

   Describe three distinct sequences of interactions with one or more functions of this system you would explore during exploratory testing of this system, based on the tour you described above. Explain the interactions you would take when executing that sequence, and why those actions fulfill the goals of that tour. These sequences should be different from each other (i.e., limited overlap).

Question 6 (Unit Testing) - 9 Points

Consider the obstacle detection method that you developed test specifications for in Question 4:

```
public String[] analyzeObstacles (List<Float> cameraData)
```

Based on your test specifications, write three JUnit-format test cases:

1. Create one test case that checks a normal usage of the method, with at least one detected obstacle.
2. Create one test case that checks a normal usage of the method, with a potential obstruction.
3. Create one test case reflecting an error-handling scenario (an exception is thrown).

Question 7 (Structural Testing) - 16 Points

This function takes an array and string. It will then remove the letters in the string from the array, one at a time, and return the array. If a letter is in the array multiple times, it will only be removed the number of times that it appears in the string (e.g., if “b” is in the array twice, and the string is “big”, then “b” will only be removed one time).

```
1.  public String[] removeLetters(String[] letters, String word) {
2.      int lettersRemoved = 0;
3.      while(word.length() != 0){
4.          for(int i = 0; i < letters.length; i++){
5.              if(letters[i].equals(word.substring(0, 1))){
6.                  letters[i] = "";
7.                  lettersRemoved++;
8.                  break;
9.              }
10.          }
11.          word = word.substring(1);
12.      }
13.      int idx = 0;
14.      String[] output = new String[letters.length - lettersRemoved];
15.      for(int i = 0; i < letters.length; i++){
16.           if(!letters[i].equals("")){
17.               output[idx] = letters[i];
18.               idx++;
19.           }
20.      }
21.      return output;
22.  }
```

For example:

- removeLetters(["s", "t", "r", "i", "n", "g", "w"], "string") → ["w"]
- removeLetters(["b", "b", "l", "l", "g", "n", "o", "a", "w"], "balloon") → ["b", "g", "w"]
- removeLetters(["d", "b", "t", "e", "a", "i"], "edabit") → [ ]

1. Draw the control-flow graph for this function. You may refer to line numbers instead of writing the full code.
2. Identify test input that will provide statement and branch coverage. You do not need to create full unit tests, just supply input for the function.

   For each input, list the line numbers of the statements covered as well as the specific branches covered (use the line number and T/F, i.e., “3-T” for the true branch of line 3). Do not use the test input from the examples above. Come up with your own input.

Question 8 (Data Flow Testing) - 12 Points

Using the same code from Question 7:

1. Identify the def-use pairs for all variables.
2. Identify test input that achieves all def-use pairs coverage.

Note: You may treat arrays as a single variable for purposes of defining DU pairs. This means that a definition to **`letters[i]`** or to array **`letters`** are both definitions of the same variable, and references to **`letters[i]`** or **`letters.length`** are both uses of the same variable.

Question 9 (Mutation Testing) - 15 Points

This function takes in an integer, and inserts duplicate digits on both sides of all digits which appear in a group of one.

```
1. public static long lonelyNumbers(int n) {
2.     String s = " " + n + " ";
3.     long a = 0;
4.     StringBuilder sb = new StringBuilder();
5.     for(int i = 1; i < s.length() - 1; i++){
6.         if(s.charAt(i) == s.charAt(i - 1) || s.charAt(i) ==
               s.charAt(i + 1)){
7.             sb.append(s.charAt(i));
8.         } else
9.             sb.append("" + s.charAt(i) + s.charAt(i) + s.charAt(i));
10.     }
11.     a = Long.parseLong(sb.toString().trim());
12.     return a;
13.   }
```

For example:

- `lonelyNumbers(4666)` → `444666`
- `lonelyNumbers(544)` → `55544`
- `lonelyNumbers(123)` → `111222333`
- `lonelyNumbers(33)` → `33`

Answer the following three questions for **each** of the following mutation operators:

- Relational operator replacement (ror)
- Arithmetic operator replacement (aor) (including short-cut operators)
- Constant for constant replacement (crp)

1. Identify all lines that can be mutated using that operator.
2. Choose **one** line that can be mutated by that operator and create **one** non-equivalent mutant for that line.
3. For that mutant, identify test input that would detect the mutant. Show how the output (return value of the method) differs from that of the original program.