College of Science – Computer Science

CSC364/M64 – Software Testing

May/June 2021

Release Time: **10:00 (Time Zone: BST)**

Deadline: **13:00 (Time Zone: BST)**

Alternative Assessment Information

- You ***MUST*** use your own copy of this assessment from Canvas. If you obtained this assessment document from a friend or elsewhere then delete this copy and use your own version from Canvas. If you experience difficulties to download the assessment, get in contact via the emails below.
- This is an open-book assessment. This means you may use your notes, textbooks, and other resources, including calculators. Copying from resources other than your notes requires referencing.
- You must submit before the deadline. Allow some spare time for technical submission issues.
- The total time for this assessment is 3 hours. This assessment is designed to be sat in a 2-hour window. An additional hour allows you time for accessing the paper, uploading your submission, and dealing with technical issues.
- It is suggested that you use Microsoft Word (or any other editor of your choice) to type your answers, then save as PDF when you are ready to submit. All submitted text (and code if present) must be word-processed, but you may include images (or photos of hand drawn images) as part of the document.
- This is an individual assessment. Under no circumstances are you to discuss any aspect of this assessment with anyone; nor are you allowed to share this document, ideas or solutions with others using email, social media, instant messaging, websites, or any other means. Your attempts at these questions must be entirely your own work. Those found to have collaborated with others will receive a mark of 0.

Special Instructions

Answer all questions.

Submission Instructions

- Please submit a **single PDF** file **named as your student number followed by the module code** (e.g. 123456-CSC789.pdf) via the submission link located on the module page in Canvas.

By submitting, electronically and/or hardcopy, you state that you fully understand and are complying with the university's policy on Academic Integrity and Academic Misconduct. The policy can be found at https://myuni.swansea.ac.uk/academic-life/academic-misconduct.

Originator(s): ***Xiuyi Fan***

In case of queries email both: xiuyi.fan@swansea.ac.uk and H.Elliott@Swansea.ac.uk

Question 1

Consider the Computational Problem **AlteredSum**:

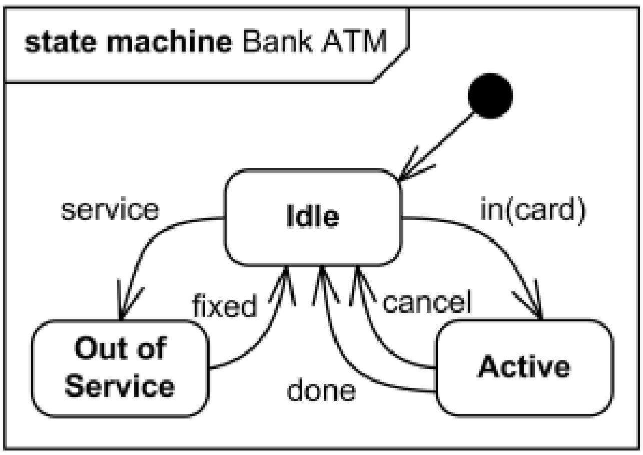
AlteredSum:
Input: integers $-10 \leq x \leq 10$ and $-5 \leq y \leq 5$
Output: integer $(x + y) * 2$

Give a test suite for boundary value testing.

[3 marks]

Question 2

Consider the Simple State Machine "Bank ATM" below. This Simple State Machine has inputs only.



1. Give the graphical representation of the Simple State Machine "Transformed Bank ATM" obtained from "Bank ATM" by applying the "insert node in transition" transformation.
2. Give an all-nodes test suite T' for "Transformed Bank ATM".

   Remark: Just give the test suite; there is no need to write a description of how you have derived it.
3. Argue briefly why T' covers for the all-nodes criterion.
4. Turn T' into an all-transitions test suite T for the Simple State Machine "Bank ATM".

[2 + 2 + 1 + 2 = 7 marks]

Question 3

Consider the assignment

$$A := (\textbf{not}\, B \textbf{ and } C) \textbf{ or } (D \textbf{ and not}\, E)$$

and the test inputs

Test Case Number	1	2	3	4
Input B	F	T	F	T
Input C	T	T	F	F
Input D	T	F	T	T
Input E	F	F	T	F

1. Give the truth table for $(\textbf{not}\, B \textbf{ and } C) \textbf{ or } (D \textbf{ and not}\, E)$ for the given four test inputs.
2. Follow the 5 step approach in order to check if your test suite complies with the MC/DC criterion. Make all 5 steps clearly visible.

[2 + 5 = 7 marks]

Question 4

Consider the following problem:

> When registering for a web-service, the user needs to provide a password. The following rules are applied for the password:
>
> - Must be between 3 and 10 characters long;
> - Must contain at least two of the following character types:
>   - Special character
>   - Lower case alpha
>   - Number
>   - Upper case alpha
> - If the password only contains two of the above, a weak symbol is displayed;
> - If the password only contains three of the above, a strong symbol is displayed;
> - If the password contains all four of the above and is longer than 8 characters, a very strong symbol is displayed;
> - If the password contains all four of the above but is 8 characters or less, a strong symbol is displayed.

Develop a test suite for this software following the decision table based testing approach by

1. Defining conditions for a binary decision table and computing the number of rules expected in the decision table;

   Note: When defining the classes, you should consider the coupling between string length and the possibility of having different character types.
2. Defining actions for an extended decision table;
3. Giving a decision table including a rule count and a comparison to the expected number of rules;
4. Providing a test suite.

[4 + 2 + 6 + 5 = 17 marks]

Question 5

Consider the following computational problem

GreatestCommonDevisor:
Input: natural numbers $x, y \geq 0$.
Output: the natural number which is greatest common divisor of x and y

and a Java program solving it:

```
public static int computeGCD(int x, int y) {
    int temp = 0;
    while (y != 0) {
        temp = y;
        y = x%y;
        x = temp;
     }
    return x;
}
```

1. Draw the program graph of `computeGCD`.
2. Give a table that lists the define/use nodes for the variables `x, y, temp`; (consider the method declaration as a node which defines `x` and `y` and consider the return statement as a computational use of `x`.)

   You should use a table with the following format:

Variable	defined at	computational use at	predicate use at

3. Give the set of definition-clear paths from every defining node v to every use of v and to the successor node of each use of v, for v in { `tmp, x, y` }.
4. Give a test suite that covers for the All-USES criterion.

   Hint: Before giving the test suite, you might want to annotate what the paths mean in terms of how often the loop needs to be executed, e.g., never enter the loop, execute the loop at least once, execute the loop at least twice, etc. You can then summarise the requirements for the loop and write test cases accordingly.

[2 + 2 + 2 + 2 = 8 marks]

Question 6

Coverage criteria are generally related in the following way:

criterion A implies criterion B
if and only if
should a test suite T cover for criterion A, then T will also cover for criterion B.

Consider the testing criteria C_0, C_p, and $C_i(k)$.

Give a proof or a counter example for the following statements:

1. C_0 implies C_p.
2. C_p implies C_0.
3. $C_i(k+1)$ implies $C_i(k)$.
4. $C_i(1)$ implies C_p.

[2+2+2+2 = 8 marks]

End of Paper