Carleton UNIVERSITY

FINAL
EXAMINATION
FALL 2017

DURATION: 3 HOURS **No. Of Students: 80**

Department Name & Course Number : **Systems and Computer Engineering SYSC SYSC 4101 / SYSC 5105**

Course Instructor: Yvan Labiche

AUTHORIZED MEMORANDA
Closed Book, No reference Material, Calculators are NOT allowed

Students MUST count the number of pages in this examination question paper before beginning to write, and report any discrepancy to a proctor. This question paper has 13 pages + cover page = 14 pages in all.

This examination question paper MAY NOT be taken from the examination room.

In addition to this question paper, students require: an examination booklet: no
Scantron Sheet: no

Name: _____

Student Number: _____

PART I:	_____	[23 marks]—General Principles
PART II:	_____	[36 marks]—Input Domain Modeling
PART III:	_____	[27 marks]—Graph Criteria
PART IV:	_____	[8 marks]—Boolean Logic Testing
PART V:	_____	[16 marks]—Drivers, Stubs and Oracles
PART VI:	_____	[14 marks]—Regression Testing
PART VII:	_____	[10 marks]—Object-Oriented Software
Total:	_____	/ 134

Instructions:

1. Answer all questions.
2. Use the number of marks allotted for each question to gauge the amount of time you should devote to that question.
3. If doubt exists as to the interpretation of any question, you are urged to make reasonable assumptions in order to proceed, but you must clearly state any assumption made.
4. **You are urged to write legibly**.

PART I. [23 marks]—General Principles

Question 1. [5 marks]
What are the key elements of the IEEE definition for Verification?

Question 2. [5 marks]
What is a fault?
What is a failure?
How do faults and failures relate to one another?

Question 3. [1 marks]
Cite one verification and validation technique other than software (dynamic) testing.

Question 4. [5 marks]
Why do we say observability is an issue in software testing?
Illustrate your answer by using a principle of object-oriented software development that hinders observability.

Question 5. [3 marks]
What do we mean when we say that white-box testing does not scale-up?

Question 6. [4 marks]
Circle the correct answer.

a) One checks, by reading through the text of use case descriptions, that all the data the system should know about are somewhat given to the system, used by the system or produced by the system in at least one use case.
This is validation This is verification

b) One is checking that the Graphical User Interface of a piece of software follows accepted GUI design standards.
This is validation This is verification

c) One checks through software executions whether the responses of the software are the ones the user/client would expect.
This is validation This is verification

d) One considers a state model and devises tests so that each state of the model is exercised at least once by the set of test cases, and then determines to what extent transitions of the state model are exercised.
This is black-box testing This is white-box testing

e) One devises test inputs from a user-defined specification, executes the code with those inputs and measures how many statements of the code have been exercised.
The all-statement criterion is a
coverage criterion selection criterion

f) One considers the body of a series of methods in a Java class and devises test inputs such that each statement of the code of the class is exercised at least once.
This is black-box testing This is white-box testing

g) One considers a state model and devises tests so that each state of the model is exercised at least once by the set of test cases, and then determines to what extent transitions of the state model are exercised.
The all-transition criterion is a
coverage criterion selection criterion

h) One has devised test inputs from a user-defined specification, has executed the code with those inputs and measures how many statements of the code have been exercised.
This is black-box testing This is white-box testing

PART II. [36 marks]—Input Domain Modeling

Question 7. [12 marks (6+6)]

A functionality you have to test takes an integer as input and behaves differently, i.e., performs different kinds of computations, depending on whether the input value is below than or equal to -10, or is strictly negative but strictly greater than -10, or is equal to 0, or is strictly positive but strictly smaller than 12, or is greater than or equal to 12 but strictly smaller than 300, or is greater than or equal to 300.

Consider the following eleven options for test inputs.

A	-10	0	12	300						
B	-30	-5	0	3	15	400				
C	-47	-8	6	12	15	317				
D	-112	-10	-5	0	9	153				
E	-10	0	10	99	305					
F	-10	0	12	259						
G	-612	-10	-2	0	3	12	15	300		
H	-403	-47	-10	-6	0	5	12	152	300	678
I	-45	-8	0	7	12	112	300	415	612	
J	-1495	-10	-8	8	12	152	298	300	412	
K	-300	-10	-5	0	7	12	212	300	1485	

1) Which of those test inputs is **a** good selection of input values to exercise partitions (equivalence classes) of the input domain, **without** focussing on boundaries?
Justify your answer, i.e., justify *why your selection is **the** good selection* and *why each of the other ones is not a good selection; you can group justifications*.

2) Which of those test inputs is **a** good selection of input values to exercise partitions (equivalence classes) of the input domain, **as well as** boundaries?
Justify your answer, i.e., justify *why your selection is **the** good selection* and *why each of the other ones is not a good selection; you can group justifications*.

Question 8. [9 marks (6+3)]

Consider the following set of characteristics and blocks.

- Characteristic 1 has two blocks labeled A and B. Block A is a base block for this characteristic;
- Characteristic 2 has three blocks labeled x, y and z. Block z is a base block for this characteristic;
- Characteristic 3 has three blocks labeled O, P and Q. Block P is a base block for this characteristic;
- Characteristic 4 has two blocks labeled 1 and 2. Block 2 is a base block for this characteristic.

1. Build a series of combinations of blocks (i.e., test case specifications) that is adequate for the **Base Block** criterion.

2. What are the test requirements for the **Pair-Wise** criterion?

Question 9. [10 marks]

Consider a function that converts currency amounts. It takes three input parameters: a *from* country name (string), a *to* country name (string) and an *amount* (real). It converts *amount* assumed to be a value in the currency of the *from* country into an amount in the currency of the *to* country. To do that it uses a web service from the Bank of Canada that provides exchange rates.

Apply Input Domain Modeling (a.k.a. Category Partition) for this function.

In the process of applying Input Domain Modeling as discussed in class, you are not asked to define base blocks or constraints for blocks.

Question 10. [5 marks]

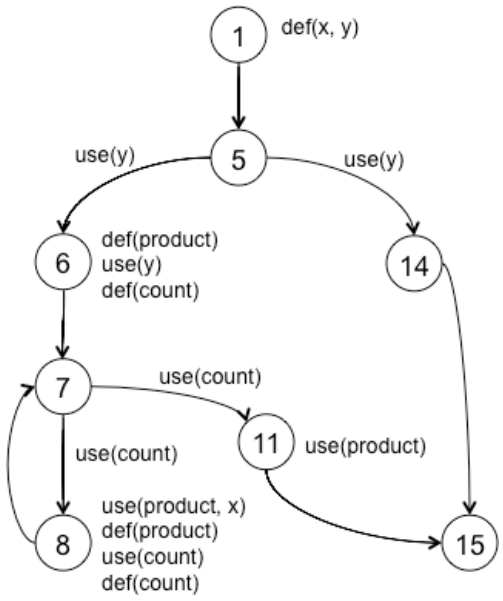
When one considers a characteristic and identifies a series of blocks for this characteristic, what are the two properties those blocks must satisfy?

PART III. [27 marks]—Graph Criteria

Question 11. [16 marks (2+2+4+6+2)]

Consider the following program that computes x^y, given integers x and y. For y<0, the program skips the computation and outputs an error message.

```
1.  begin
2.    int x, y;
3.    int product, count;
4.    input(x,y);
5.    if (y>=0) {
6.      product=1;
7.      count=y;
8.      while(count>0) {
9.        product=product*x;
10.       count=count-1;
11.     }
12.     output(product);
13.   } else {
14.     output("Wrong input.");
15.   }
16. end
```



a) What are the test requirements for the All-Edges criterion? List them.

b) Create test path(s) to satisfy the All-Edges criterion.

c) Find test inputs to execute the test paths you identified in the previous question

d) Fill in the table below with information about definitions and uses of variables (i.e., locations where all the variables of the program are defined and used) as well as DU-paths for the identified definition-use pairs. You will not consider line number 3 as a defining location for *product* and *count*.

The table is filled with a fictitious example whereby (1) a definition of variable *var* appears in node 10 can reach using node 12 with only one DU-path 10-11-12, (2) the same definition can reach a use in edge 14-15 with two different DU-paths.

Note that the number of rows in the table is not an indication of the amount of information that must be provided in a correct answer to this question.

Variable	Definitions	Uses	DU-paths
var	10	12	10-11-12
		14-15	10-11-12-13-14-15
			10-11-14-15

e) Create test path(s) to satisfy the All-Uses criterion.

Question 12. [5 marks]

Create the Control Flow Graph corresponding to the following program, using the condensed form and numbering nodes after line numbers. (As much as possible, place the true branch of alternatives or loops on the left hand side and the false branch on the right hand side.)

```
1.  begin
2.    int x, y;
3.    int product, request;
4.    int exp=1, fact=2, exit=3;
5.    get_request(request);
6.    product = 1;
7.    while (request <> exit) {
8.      if (request == exp) {
9.        input(x, y);
10.       while (count>0) {
11.         product=product*x; count--;
12.       }
13.     } else if (request==fact) {
14.       input(x); count=x;
15.       while(count>0) {
16.         product=product*count; count--;
17.       }
18.     } else if(request==exit) {
19.       output("Thank you.");
20.       break;
21.     }
22.     output(product);
23.     get_request(request);
24.   }
25. end
```

Question 13. [6 marks (2+2+2)]

1. What does the program of the previous question do?
2. In the program of the previous question, using the control flow graph you created, what is the path of the graph taken by the program if the input provided by the user (through the call to `get_request()`) is neither 1, nor 2, nor 3?
3. Would the output of the program make sense in the input you identified earlier is actually used?

PART IV. [8 marks]—Boolean Logic Testing

Question 14. [8 marks]

Consider the following Boolean expression: `Z = ( A and D and not(C) ) or ( A and B ).`
Determine test inputs that satisfy the Restricted Active Clause Criterion.

PART V. [16 marks]—Drivers, Stubs and Oracles

Question 15. [6 marks]
The oracle assumption states that testers are routinely able to determine whether or not the test output is correct. A non-testable software is a software for which this assumption does not hold. What are the two situations that define a non-testable software, i.e., that make the assumption above untrue?

Question 16. [5 marks]
Suppose you have created a test driver to execute one test case, that the test driver includes an oracle (you will assume the unit under test is testable), and that to perform the test you needed to simulate parts of the environment with stubs. If the test case this driver executes fails, what can you conclude regarding the coming debugging task? Justify.

Question 17. [5 marks]
Your colleague and you have to test two classes, called A and B. You want to split the work: your colleague is in charge of testing class A and you are in charge of testing class B. Both classes A and B use the services of a piece of hardware that is not yet available. Your colleague argues that because there is only one hardware equipment both A and B use, there should only be one stub simulating this equipment to be used when testing both classes A and B.
Do you agree with this statement? Why?

PART VI. [14 marks]—Regression Testing

Question 18. [6 marks (2+2+2)]

a) What is an Obsolete test case?

b) What is a Re-testable test case?

c) What is a Reusable test case?

Question 19. [4 marks]

Explain why there is a need for minimization or prioritization in the context of regression testing?

Question 20. [4 marks]

Consider the following scenario: You are working for a consulting company, specializing in software testing, and are discussing with a client. The client wants you to help them conduct some regression testing activity on the current version of their software. The client explains that they have an initial version of the software (say, v1) with a test suite that has passed, and that they just finished developing a second version of the software (say, v2) that they want to regression test. What is the *very first thing* you need to ensure with the client to identify whether regression testing is at all feasible in this context? Justify.

PART VII. [10 marks]—Object-Oriented Software

Question 21. [5 marks]
Explain why inheritance sometimes leads to additional testing effort.

Question 22. [5 marks]
Explain what is “Inheritance Context Coverage”.