Midterm exam 2018, questions

Software Quality Assurance (University of Ottawa)

Question 1 (15 marks)

Suppose a C function find_tax used as part of a software for income tax calculation. The header of the function is as follow:

```
double find_tax(double taxable_income,
                int age,
                int number_of_dependents)
```

Each parameters (*taxable_income*, *age* and *number_of_dependents*) must be greater or equal to 0.
find_tax returns the *net income tax* (NET_TAX) according to following calculations:

- NET_TAX = BASE_TAX - CREDITS
- BASE_TAX is found as follow
  - if *taxable_income* ≤ 42707
    BASE_TAX = *taxable_income* * 15%
  - if 42707 < *taxable_income* ≤ 85414
    BASE_TAX = 6406 + (*taxable_income* - 42707) * 22%
  - if 85414 < *taxable_income* ≤ 132406
    BASE_TAX = 15802 + (*taxable_income* - 85414) * 26%
  - if *taxable_income* > 132406
    BASE_TAX = 28020 + (*taxable_income* - 132406) * 29%

  CREDITS = 10822 + AGE_CREDIT + DEPENDANTS_CREDIT
  - if *age* ≥ 65, AGE_CREDIT = 6720,
    otherwise AGE_CREDIT = 0
  - if *number_of_dependents* * 4300 ≤ 12900,
    DEPENDANTS_CREDIT = *number_of_dependents* * 4300
    otherwise DEPENDANTS_CREDIT = 12900

i) Using the Equivalence Class Partitioning approach, partition find_tax inputs in Equivalence Classes (**8 marks**).

ii) Provide a minimum number of test cases covering ***all*** valid equivalence classes, and ***two*** invalid equivalence class listed in ***i)***. Use boundary values when possible (7 **marks**).

Test Case number	***Test Data***	***Equivalence Classes***

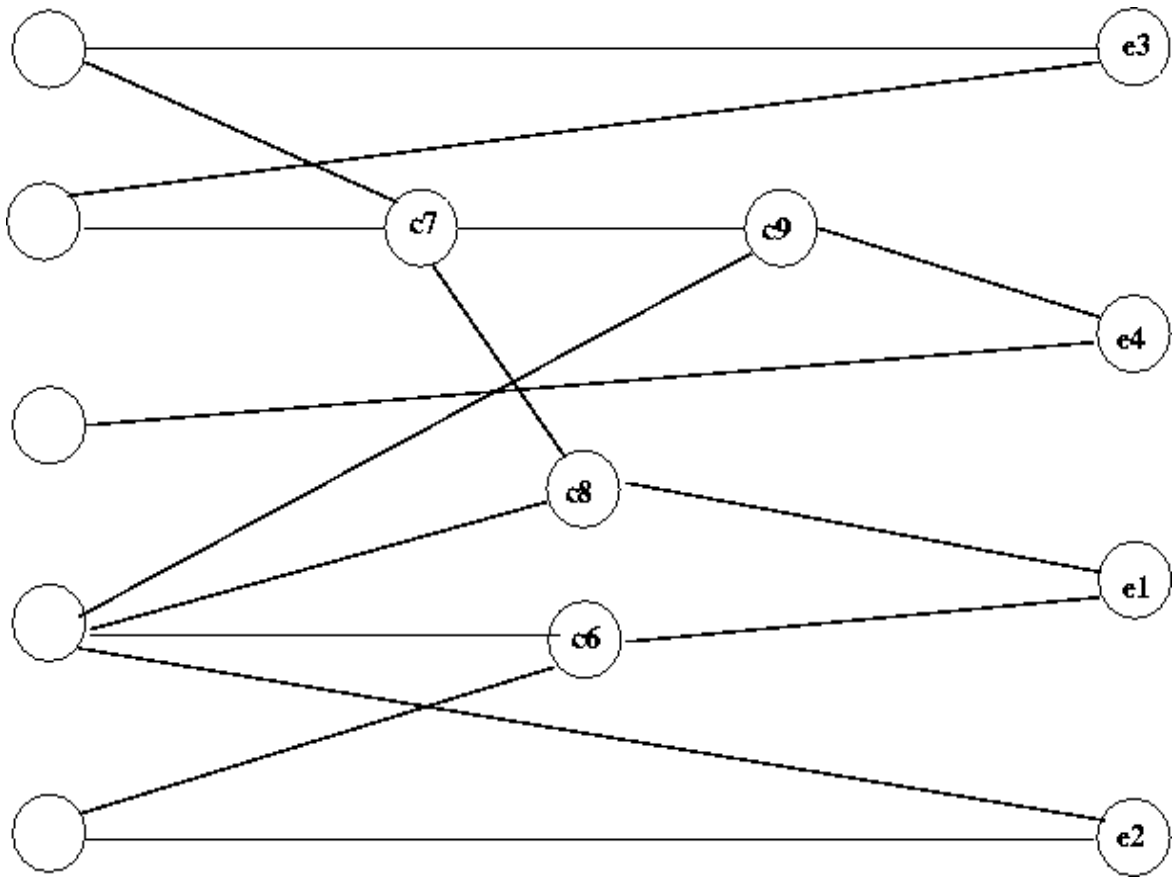
Question 2 (15 marks)

Consider the following description of an automated tomato/tulip dispensing machine:
If the user pushes button A on the machine, (s)he receives a tomato, unless the machine is out of tomatoes, in which case the message "choose B or C" is displayed for two seconds. If the user pushes button B and the machine is not out of tulips, (s)he receives a tulip. If the user pushes button C, (s)he receives a mild electric shock. Also, if the user pushes button B and the machine is out of tulips, (s)he gets a tomato, unless the machine is also out of tomatoes, in which case (s)he receives a mild electric shock.
Consider that only one button can be pushed at the same time .

The following causes and effects have been identified for this problem:

- c1: user pushes button A
- c2: user pushes button B
- c3: user pushes button C
- c4: machine is out of tomatoes
- c5: machine is out of tulips
- e1: user receives a tomato
- e2: machine displays message "choose B or C" for two seconds
- e3: user receives a tulip
- e4: user receives a mild electric shock

i) Complete the following causes-effects graph according to the automated tomato/tulip dispensing machine description (**6 marks**).



ii) Give a boolean formula corresponding to effects ***e1*** (user receives a tomato) and ***e4*** (user receives a mild electric shock) (***4 marks***).

iii) Give variants for effect ***e4*** using the ***each-condition/all-conditions*** approach (***4 marks***)

Question 3 (12 marks)

Consider the following method.

```
  int computeVal(int x, int y) {
1    z = 0;
2    if (x > y) {
3       if (y > 5) {
4         z = x;
5       } else {
6         z = y;
7       }
8    }
9    if ((z == 4) || (z < 0)) {
10      z = z * x;
11   }
12   if (x <= y) {
13      z = z + 1;
14   } else {
15      z = z – 1;
16   }
17  return z;
18  }
```

i) Provide a simplified flowchart for method ***computeVal*** annotated with data flow actions **(4 marks)**

ii) Provide a *minimum* number of test cases such that 100% instruction coverage of method ***computeVal*** is achieved. (**4 marks**).

Test Case	***Test Data***	***Path Covered***

iii) Provide a *minimum* number of test cases that satisfy the minimum loop coverage of method ***computeVal***. You can re-use test cases from ii) by referring to them. (**4 marks**).

Test Case	***Test Data***	***Path Covered***