ECE 322

SOFTWARE TESTING AND MAINTENANCE

Fall 2021

Assignment #5

SOLUTIONS

Due date: Monday, November 22, 2021 by 11:00 PM

Total: 40 points

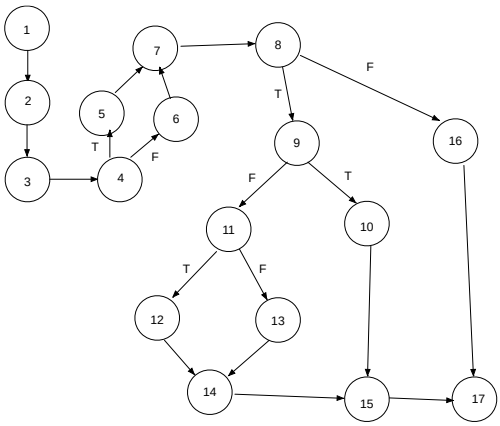
10 points

1.For the code shown below, draw a control flow graph and determine its cyclomatic complexity. Consider two situations:
(a)when compound decisions (such as those shown in lines 4, 9, and 11) are treated *en bloc*
(b)when the individual conditions in the compound decisions are treated separately.

```
1 output ("Enter 3 integers")
2 input (a, b, c)
3 output("Side a,b c: ", a, b, c)
4 if (a < b) and (b < a+c) and (c < a+b)
5 then isTriangle ← true
6 else isTriangle ← false
7 fi
8 if isTriangle
9 then if (a = b) and (b = c)
10 then output ("equilateral")
11 else if (a ≠ b ) and ( a ≠ c ) and ( b ≠ c)
12 then output ("scalene")
13 else output("isosceles")
14 fi
15 fi
16 else output ("not a triangle")
17 fi
```

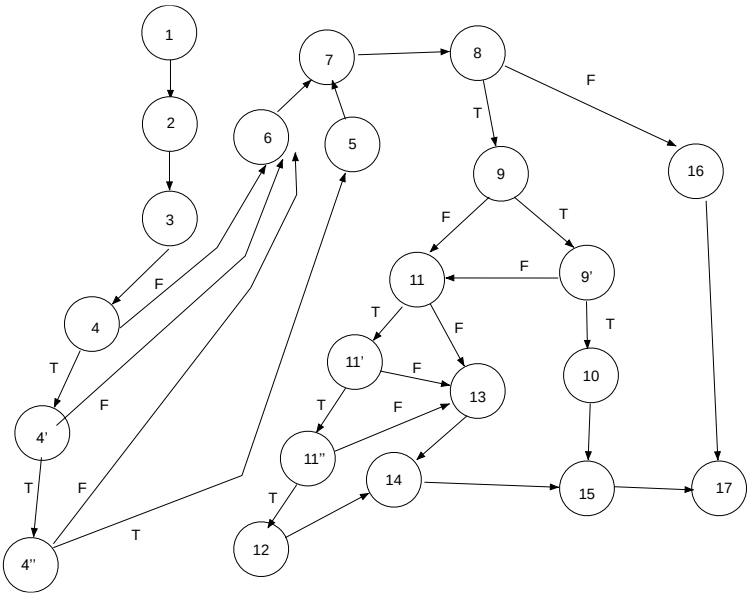
Solution

In case compound decisions are treated *en bloc*, the control flow graph is shown below.



By counting the number of binary decision boxes, the cyclomatic complexity is 5 (number of regions of the planar graph).

If the individual conditions are considered, the cyclomatic complexity of the corresponding control flow graph (shown below) is 10.



10 points

2.For the piece of code shown below show *def-clear* paths for the variables *tv*, *av*, and *sum*.

```
public static double ReturnAverage(int value[],
                          int AS, int MIN, int MAX){
  /*
   Function: ReturnAverage  Computes the  average
   of all  those  numbers in the  input array  in
   the  positive  range  [MIN, MAX]. The  maximum
   size  of the array is AS. But, the  array size
   could be smaller than AS in which case the end
   of input is represented by -999.
  */
     int i, ti, tv, sum;
     double av;
     i = 0; ti = 0; tv = 0; sum = 0;
     while (ti < AS && value[i] != -999) {
         ti++;
         if (value[i] >= MIN && value[i] <= MAX) {
            tv++;
            sum = sum + value[i];
         }
         i++;
     }
     if (tv > 0)
        av = (double)sum/tv;
     else
        av = (double) -999;
     return (av);
}
```

Solution

```
public static double ReturnAverage(int value[],
                           int AS, int MIN, int MAX){
  /*
   Function: ReturnAverage  Computes the  average
   of all  those  numbers in the  input array  in
   the  positive  range  [MIN, MAX]. The  maximum
   size  of the array is AS. But, the  array size
   could be smaller than AS in which case the end
   of input is represented by -999.
  */
     int i, ti, tv, sum;
     double av;
   1 i = 0; ti = 0; tv = 0; sum = 0;
   2 while (ti < AS && value[i] != -999) {
       3 ti++;
       4 if (value[i] >= MIN && value[i] <= MAX) {
           5 tv++;
           6 sum = sum + value[i];
         }
       7 i++;
     }
   8 if (tv > 0)
       9 av = (double)sum/tv;
     else
       10 av = (double) -999;
  11 return (av);
}
```

Recall that for variable *x* a path (*i*, *n*1, *n*2, …,n_j, *j*) is a def-clear path if it starts at node *i* where variable *x* has been defined and terminates at node *j* and there are no definitions of variable *x* at in-between nodes (*n*1, *n*2, …,n_j, *j*).
By analyzing the resulting numbering, we enumerate the paths for the variables

variable *tv:* 1□5 (compute use), 1□9 (compute use),1□8 (predicate use), 5□ 8 (predicate use)

variable *av:* 9□11 (compute use), 10□11 (compute use)

variable *sum:* 1□6 (compute use), 1□9 (compute use)

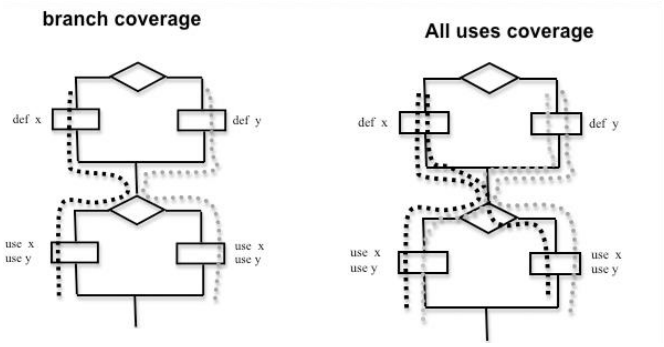
10 points

3. Write a function in Python such that the *all-uses* coverage criterion produces more test cases than the *branch* coverage criterion.

Solution

```
def (x, y):
    if x>y:
        x = 1
    else:
        y = 1
    if x*y<0:
        z = -x*y
    else:
        z = x*y
    return z
```

A code leads to a control flow graph shown below



10 points

4.Using the modified condition/branch coverage criterion, propose test cases for the following expression

$$(a \,||\, b) \;\&\&\; (\text{not}(c) \,||\, \text{not}(d))$$

Note that the test set is not unique; show all possibilities.

Solution

The table below displays all combinations of variables

	a	*b*	*c*	*d*	*z*
1	F	F	F	F	F
2	F	F	F	T	F
3	F	F	T	F	F
4	F	F	T	T	F

5	F	T	F	F	T
6	F	T	F	T	T
7	F	T	T	F	T
8	F	T	T	T	F
9	T	F	F	F	T
10	T	F	F	T	T
11	T	F	T	F	T
12	T	F	T	T	F
13	T	T	F	F	T
14	T	T	F	T	T
15	T	T	T	F	T
16	T	T	T	T	F

Test cases:
For *a:* 1-9, 2-10, 3-11
For *b*: 1-5, 2-6, 3-7
For *c*: 6-8, 10-12, 14-16
For *d*: 7-8, 11-12, 15-16

The number of test cases is 5, for instance, 2, 10, 6, 11, 12; 3,11, 7,6, 8; 2,10,6,8,7; 3,11,7,10,12.