SE 333 Software Testing

Assignment 6: Test coverage Design

Due date: May 10, 2021, 11:59pm

Late penalties: 1% late penalty per day for up to 7 days. No late submission will be accepted after that.

1. **Given the following method in Java...**

```
/**
* Returns the subset of ints that are greater than the provided
* target value
* @param a : an array of ints to search
* @param target : the value to search for
* @return the subset
*/
public static int[] greaterThan(int[] a, int target) {
 List<Integer> found = new ArrayList<>();
 if (a != null && a.length > 0) {
   for (int i=0; i < a.length; i++) {
     if (a[i] > target) {
       found.add(a[i]);
     }
   }
   return found.stream().mapToInt(Integer::intValue).toArray();
 }
 throw new IllegalArgumentException("Input array is null or empty");
}
```

   a. Design a test suite with the fewest number of test cases that satisfies the statement test criterion.

      Define the test suite as one or more literal arrays:

      T1 = [x, y, …], n

      where the values in the array are actual integer values.

      [] can be used to indicate an empty array

The word ‘null’ by itself (not inside brackets) can be used to indicate … null.

Notice that a test case requires 2 values:
1. an array to search
2. an int to search for

b. Design a test suite with the fewest number of test cases that satisfies the branch-condition test criterion.
Define this test suite using the same technique as above

2. Consider the following code fragment in C:

```
int max_size = 10;
int count = 0;
for (n = 0;  n < max_size && (c = getc(yyin)) != EOF’; ++n) {
        count++;
        if (count < 10) { . . .}
```

		A	B	C	
ID	input	n < max_size	c = getc(yyin) != EOF	count < 10	Result
1	n = 5; c = ‘X’;count=7	True	True	True	True
2					
. . .					

a. Derive a set of test cases that satisfy the compound condition adequacy criterion with respect to this code. Document your test cases by extending the table above.

b. Derive a subset of test cases from the table above that satisfy the modified condition (MC/DC) adequacy criterion with respect to the loop. Express these test cases using their IDs and the clause that they test:

A = 1 and 3 (means test case 1 and test case 3 combine to satisfy MC/DC test adequacy for condition A)
B = . . .

3. Consider the following loop statement in C:

```
Int max_size = 10;
for (n = 0;  n < max_size && (c = getc(yyin)) != EOF && c != '\n'; ++n)
        buf[n] = (char) c;
```

		A	B	C	
ID	input	n < max_size	c = getc(yyin) != EOF	c != '\n'	Result
1	n = 5; c = 'X'	True	True	True	True
2					
. . .					

a. Notice this problem is very similar but not identical to problem 3.

b. Derive a set of test cases that satisfy the compound condition adequacy criterion with respect to the loop. Document your test cases by extending the table above. Do not provide any unrealistic test cases

c. Calculate the MC/DC subset as in problem 2. How is this subset different from the set of test cases defined in 3b above. You do not need to actually display the "A = 1 and 3 …" list if you don't want to, just answer the question. If showing the pairs list helps you express your answer then by all means, do show it.

4. Let us consider the following if statement in Java

```
// assume parseArray.length is 4
if ( pos < parseArray.length &&
        (parseArray[pos] == '{' ||
        parseArray[pos] == '}' ||
        parseArray[pos] == '|') ) {
```

```
            continue;
        }
```

Derive a set of test cases and show that it satisfies the modified condition (MC/DC) adequacy criterion for this statement. For brevity, abbreviate each of the basic condition as follows:

Room	pos < parseArray.length
Open	parseArray[pos] == '{'
Close	parseArray[pos] == '}'
Bar	parseArray[pos] == '|'

Provide the answer in 2 parts:

1. a table showing the test cases you have designed, similar to this one:

ID	input	Room	Open	Close	Bar	Result
1	pos=1; pa[pos]='a'	True	False	False	False	False
2						
. . .						

2. List of pairs of test cases similar to what you did in question 2:

Room = 1 and 3

You may want to create a table for compound condition adequacy first but you do not have to deliver it if you don't want to.

Deliverable:
A document containing your solutions to these problems in one of these formats:
PDF
Word
Plain text
Excel

No photocopies of solutions written on paper.