SE465 Final W2016

1.
   a) [Not covered?]
   b) [Not covered]
   c) [Not covered?]
   d) Data races, deadlock. Sanitizers (eg tsan) can detect these bugs.
   e) a AND b (the pairs (true, false) and (false, true) satisfy GACC, but the predicate evaluates to false for both)

2.
   a) The warning states that we deference a value (other) that could be null. It is a true bug, since we never check that other is not null before accessing its size property.
   b) Summary, description (with expected and actual results), build date/platform
   c) Summary: StrBuilder.equalsIgnoreCase method dereferences possible null object
      Description: In the equalsIgnoreCase method of StrBuilder.java, on line 12, the other object's size property is accessed without checking that it is not null.

      Test case:

```
@Test
void test() {
   StrBuilder first = new StrBuilder();
   StrBuilder second = null;
   assertEquals(first.equalsIgnoreCase(second), false);
}
```

      Expected output: passes test case without throwing
      Actual output: NullPointerException

3. a)

$$\begin{tikzcd}
 & & {} \arrow[d] & & \\
 & & a:2\text{-}3 \arrow[d] & & \\
 & & b:4\text{-}6 \arrow[dll] \arrow[d] \arrow[drr] & & \\
c:7\text{-}8 & & d:9\text{-}12 \arrow[dl] \arrow[dr] & & h:20\text{-}22 \arrow[ull, bend right] \\
 & e:13\text{-}14 \arrow[dr] & & f:15\text{-}16 \arrow[dl] & \\
 & & g:18\text{-}19 \arrow[uuu, bend right=80, looseness=2] & &
\end{tikzcd}$$

b) Pretend this is a CFG.
paperFilterIsClean is defined in nodes a, e, f
paperFilterIsClean is used in node d

c) abdfgbc

d) The test requirement that uncovers the failure is f (needs to contain node f)
Criteria PPC, EPC, EC, NC, ADUPC, AUC uncover the failure.

e) returnedStatus = 'coffee cup is empty'

4. (pancakeSort, pancakes, 1) -> (findFlipSpot, pancakes, 27)
(findFlipSpot, the_spot, 28) -> (pancakeSort, the_spot, 10)
(pancakeSort, i, 8) -> (findFlipSpot, i, 26)

5\.

a) Total statement coverage: TC1, TC5, TC2, TC6, TC4, T3
Additional statement coverage: TC1, TC2, TC3, TC4, TC5, TC6
b) Additional statement coverage is better. It covers all statements after the 3rd test case, while total statement coverage doesn't cover all statements until all 6 test cases have run.
c) Since coverage is collected from the last execution, statements may have changed in the code that change the coverage of each test case.

6\.

a) $a \land (b \lor c)$
b) $p_a = p_{a=true} \oplus p_{a=false} = (b \lor c) \oplus false = b \lor c$
$p_b = p_{b=true} \oplus p_{b=false} = a \oplus a \land c = a \land \neg c$
$p_c = p_{c=true} \oplus p_{c=false} = a \oplus a \land b = a \land \neg b$

	a	b	c	p	p_a	p_b	p_c
1	T	T	T	T	T	F	F
2	T	T	F	T	T	T	F
3	T	F	T	T	T	F	T
4	T	F	F	F	F	T	T
5	F	T	T	F	T	F	F
6	F	T	F	F	T	F	F
7	F	F	T	F	T	F	F
8	F	F	F	F	F	F	F

c) a: (1, 5), (1, 6), (1, 7), (2, 5), (2, 6), (2, 7), (3, 5), (3, 6), (3, 7)
b: (2, 4)
c: (3, 4)
d) (2, 3, 4, 5)
e) a: (1, 5), (2, 6), (3, 7)
b: (2, 4)
c: (3, 4)

7\.

a) The first element in the array is less than 0. Necessary: If the first element in the array is greater than zero, then in the first iteration of the for loop max is set to that value, and we never have an invalid state. Sufficient: If the first element in the array is less than zero, after one iteration of the for loop, we have that max=0, which is an incorrect state since the element we iterated is less than 0.
b) a = [-1], aLength = 1, expected output = -1. The output of the mutant on this test case will be 0, which is incorrect.

8.

a) Valid:
   - All three input integers are different from each other.
   - The sum of the three input integers is greater than 10.

   Invalid:
   - The sum of a and b is less than c.
   - Only one input value is provided.

b)

5	Valid (3)	2,3,4	Scalene triangle
6	Valid(4)	10,10,10	Equilateral triangle
7	Invalid (3)	1,1,10	Invalid inputs
8	Invalid (4)	1	Invalid inputs

9.

a)

Test Case No.	**Size**	**Energy Source**	**Where Made**
1	hatch-back	Hybrid	North America
2	2-door	Hybrid	North America
3	4-door	Hybrid	North America
4	hatch-back	Gas	North America
5	hatch-back	Electric	North America
6	hatch-back	Hybrid	Europe
7	hatch-back	Hybrid	Asia
8	hatch-back	Hybrid	Other

b) We need $3 \cdot 3 \cdot 4 = 36$ test cases.