# HW #4: Due May $5^{th}$ 23:59

1. Generate test cases that covers all possible execution paths of the triangle program through depth first search (DFS) traversal.
   1. Assume that initial test case is (1,1,1)
   2. Write down a current executed symbolic path condition $\phi$ and a next symbolic path condition $\psi$ obtained through DFS traversal
   3. Write down a LIA SMT formula for the next symbolic path condition $\psi$
   4. Solve the LIA SMT specification through Z3
      1. If UNSAT, try to negate another  branch in a DFS order
      2. If SAT, record the solution as next input and repeat from Step 2 until all paths are covered/

You have to submit

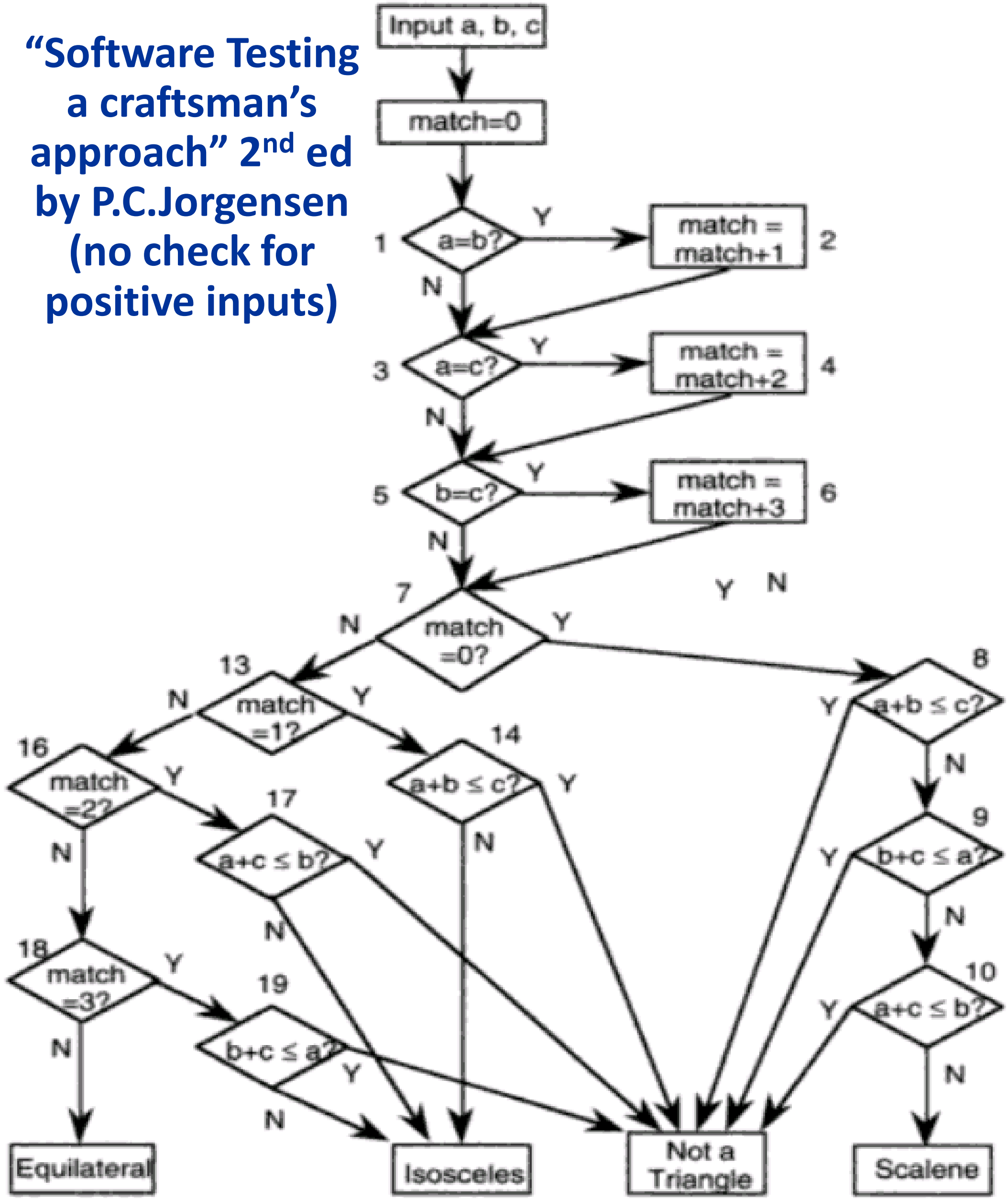
- a completed table in page 4 that contains test cases and all executed symbolic path conditions and next PCs
- a completed execution tree whose leaves are marked with TCs or UNSAT in pg 4
- All LIA SMT specifications for $\psi$s and their solutions by Z3 (submit both hardcopy and softcopy that should be sent to hongshin@gmail.com)

Note 1. CREST does not work in a DFS order, but a reverse DFS order

Note 2. You can obtain an executed symbolic path condition by executing the instrumented triangle program by `crestc` with an `input` file that contains `a`, `b`, and `c` in a string format and running `print_execution`.

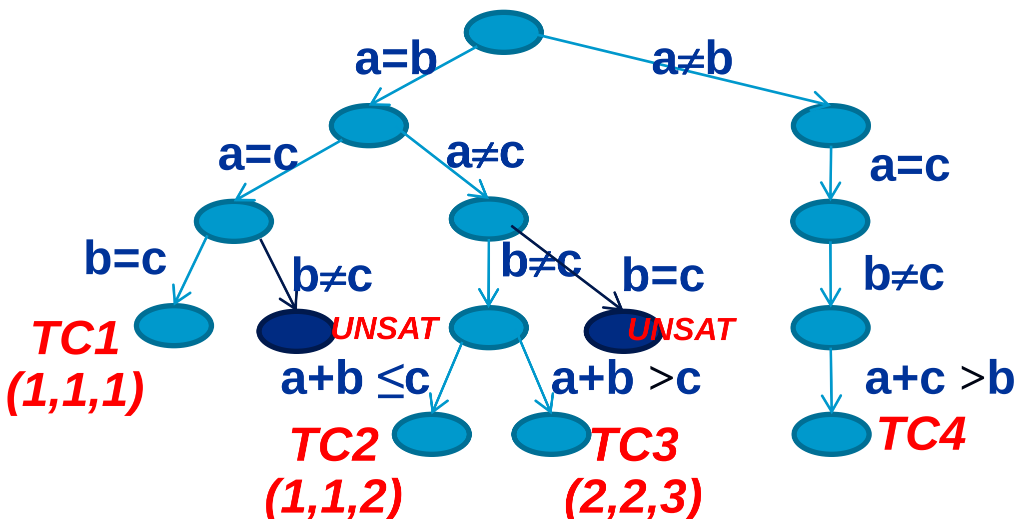
```c
#include <crest.h>
int main() {
   int a,b,c, match=0;
   CREST_int(a); CREST_int(b); CREST_int(c);
   // filtering out invalid inputs
   if(a <= 0 ) exit();  if(b <= 0 ) exit();  if(c <= 0 ) exit();
   printf("a,b,c = %d,%d,%d:",a,b,c);
   //0: Equilateral, 1:Isosceles,
   // 2: Not a traiangle, 3:Scalene
   int result=-1;
   if(a==b) match=match+1;
   if(a==c) match=match+2;
   if(b==c) match=match+3;
   if(match==0) {
      if( a+b <= c) result=2;
      else if( b+c <= a) result=2;
      else if(a+c <= b) result =2;
      else result=3;
   } else {
      if(match == 1) {
         if(a+b <= c) result =2;
         else result=1;
      } else {
         if(match ==2) {
            if(a+c <=b) result = 2;
            else result=1;
         } else {
            if(match==3) {
               if(b+c <= a) result=2;
               else result=1;
            } else result = 0;
         } }}
   printf("result=%d\n",result); }
```

**"Software Testing a craftsman's approach" 2nd ed by P.C.Jorgensen (no check for positive inputs)**



# Concolic Testing the Triangle Program

| Test case | Input (a,b,c) | Executed symbolic path condition (PC) $\phi$ | Next PC $\psi$ | Solution for the next PC $\psi$ from SMT solver |
|---|---|---|---|---|
| 1 | **1,1,1** | $a=b \wedge a=c \wedge b=c$ | $a=b \wedge a=c \wedge b\neq c$ | Unsat |
| | | | $a=b \wedge a\neq c$ | 1,1,2 |
| 2 | 1,1,2 | $a=b \wedge a\neq c \wedge b\neq c \wedge a+b \leq c$ | $a=b \wedge a\neq c \wedge b\neq c \wedge a+b>c$ | 2,2,3 |
| 3 | 2,2,3 | $a=b \wedge a\neq c \wedge b\neq c \wedge a+b >c$ | $a=b \wedge a\neq c \wedge b=c$ | Unsat |
| | | | $a\neq b$ | 2,1,2 |
| 4 | 2,1,2 | $a\neq b \wedge a=c \wedge b\neq c \wedge a+c>b$ | $a\neq b\wedge a=c \wedge b\neq c \wedge a+c \leq b$ | 2,5,2 |



**2.** Prove the following sequents in predicate logic:

(a) $\forall x \, \forall y \, \forall z \, (S(x, y) \land S(y, z) \to S(x, z)), \ \forall x \, \neg S(x, x) \vdash \forall x \, \forall y \, (S(x, y) \to \neg S(y, x))$

(b) $\forall x \, (P(x) \lor Q(x)), \ \exists x \, \neg Q(x), \ \forall x \, (R(x) \to \neg P(x)) \vdash \exists x \, \neg R(x)$

(c) $\forall x \, (P(x) \to (Q(x) \lor R(x))), \ \neg \exists x \, (P(x) \land R(x)) \vdash \forall x \, (P(x) \to Q(x))$

(d) $\exists x \, \exists y \, (S(x, y) \lor S(y, x)) \vdash \exists x \, \exists y \, S(x, y)$

(e) $\exists x \, (P(x) \land Q(x)), \ \forall x \, (P(x) \to R(x)) \vdash \exists x \, (R(x) \land Q(x))$.

**3.** Prove the following sequents in predicate logic using both natural deduction and semantic tableau

(a) $S \to \exists x \, Q(x) \vdash \exists x \, (S \to Q(x))$

(b) $\exists x \, P(x) \to S \vdash \forall x \, (P(x) \to S)$

(c) $\forall x \, P(x) \to S \vdash \exists x \, (P(x) \to S)$

(d) $\forall x \, (P(x) \lor Q(x)) \vdash \forall x \, P(x) \lor \exists x \, Q(x)$

(e) $\forall x \, \exists y \, (P(x) \lor Q(y)) \vdash \exists y \, \forall x \, (P(x) \lor Q(y))$.