HW4: Due Nov 24th 23:59

1. Describe test cases to reach full path coverage of the triangle program by completing the path condition table below. Also, draw the complete execution tree showing executed path conditions.
   - Assume that the initial test case is given as 1,1,1
   - You should use the DFS algorithm.
   - Note that CREST uses *reverse*-dfs search heuristics in fact. Thus, your solutions will be different from what CREST generated

Test case	Input (a,b,c)	Executed path conditions (PC)	Next PC	Solution for the next PC
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. Testing Busybox expr

- Use `Makefile` and `cil` package we distribute to compile busybox `expr` with CREST. Also, use Busybox 1.17.0.
- For busybox `expr`, generate 10,000 test cases through the DFS search strategy. You are requested to modify `expr.c` to create test cases through CREST and feed those generated test cases to `expr`
  1. Describe which variables are declared symbolically and how
     - How long is EXPR string, too
  2. Describe how you modified the target code to improve branch coverage
  3. Create 10,000 test cases in files (i.e.,tc1, tc2,… tc10000)
  4. Measure the final branch coverage reported by CREST (i.e., MC/DC coverage)
     - You can do this by analyzing `branch` and `coverage` output file
  5. Apply the 10,000 test cases (tc1,…tc10000) to `expr` and measure the branch coverage reported by `gcov`
     - For this task, you should not use CREST. Use original `Makefile` instead of what we distribute
     - You may build a shell script to execute busybox expr 10,000 times with 10,000 test cases. Also, you may rename 10,000 test cases to a single name at each testing iteration (i.e., mv tc%d tc)

Detailed Experiment Setting for Testing Busybox expr

There are two steps to enable CREST run for busybox 1.17.0.

1. Install a new version of CIL, before compile busybox with CREST.
   1.1 Unzip the attached cil.tar.bz2 to local folder
   1.2 Copy the cil/ directory to /usr/local (cilly will be in /usr/local/cil/bin/cilly).
   or copy the cil/ directory to somewhere you want
   1.3 type ./configure && make && make install.

2. Modify Makefile in busybox.
   1.1 Replace Makefile in busybox by attached Makefile.
   1.2 Modify cilly path in CC to the path of cilly in your computer.
   1.3 Modify -I/usr/local/include in CFLAGS to include crest.h in your computer.
   1.4 Modify -L/usr/local/lib in LDFLAGS to includes libcrest.a in your computer.