HW6 writeup

1. Rewrite the program in SSA form
   See ssa.txt

2. Translate to SMT
   See bounds-check_p2.smt

3. Bounds checks
   See bounds-check_p3.smt and output_p3

4. Interpretation
   See bounds-check_p4.smt and output_p4|
   a. Line 7 – sat:
      int x = 6;
      int data[7]; // any length 7 int array
      int N=7;
      f(x, data, N) will crash at line 7 because the code tries to access data[7]
   b. Line 9 – unsat:
      no crash
   c. Line 12 – sat:
      int x = 1048580;
      int data[1114113];
      int N = 1114113;
      The satisfying assignment reported will let f(x, data, N) crash at line 7 before reaching line 12 because the code tries to access data[2097155].
      After I constrain the first data access at line 7 to be in bound, the assertion at line 12 becomes unsat (see bounds-check4_b.smt and output_p4b).
   d. Line 14 – sat:
      int x = 1;
      int data[1073741822];
      data[0]= 1073741825;
      data[1]= 2149580831;
      int N = 1073741822;
      f(x, data, N) will crash at line 14 because the code try to access data[7] because the code tries to access data[2149580831].

5. Find the smallest value of N that can cause out-of-bound access in line 7
   See bounds-check_p5.smt and output_p5
   The smallest N is 7 that cause out-of-bound access at line 7.