Consider the following Pascal program which calculates the greatest common divisor of two integers \( m \) and \( n \) and returns the result in \( a \).

**ENTRY:** \( m > 0 \) & \( n > 0 \)

**EXIT:** \( a = (m, n) \)

procedure gcd (m,n:integer; var a:integer);
var b:integer;
begin
  a:=m;
  b:=n;
  assert ((a,b)=(m,n) & a>0);
  while a<>b do
    if a>b then a:=a-b
    else b:=b-a
  end;
end;

1. You are to prove that this program is consistent with its entry and exit specifications using Backwards Substitution. Be sure to show all justifications on every line, as I did in class.

2. Use unisex to symbolically execute the gcd program. There is a copy of the program, with the assertions formatted for unisex, in CS266/PASCAL. You need only turn in the VCs that are generated.

3. How do the VCs of your proof in question 1 compare to what had to be proved in the symbolic execution proof of gcd in question 2?

EXTRA: If you care to, you can generate a proof of the gcd program using Hoare’s axiomatic approach. How do the Lemmas of this proof compare to the VCs of questions 1 and 2?