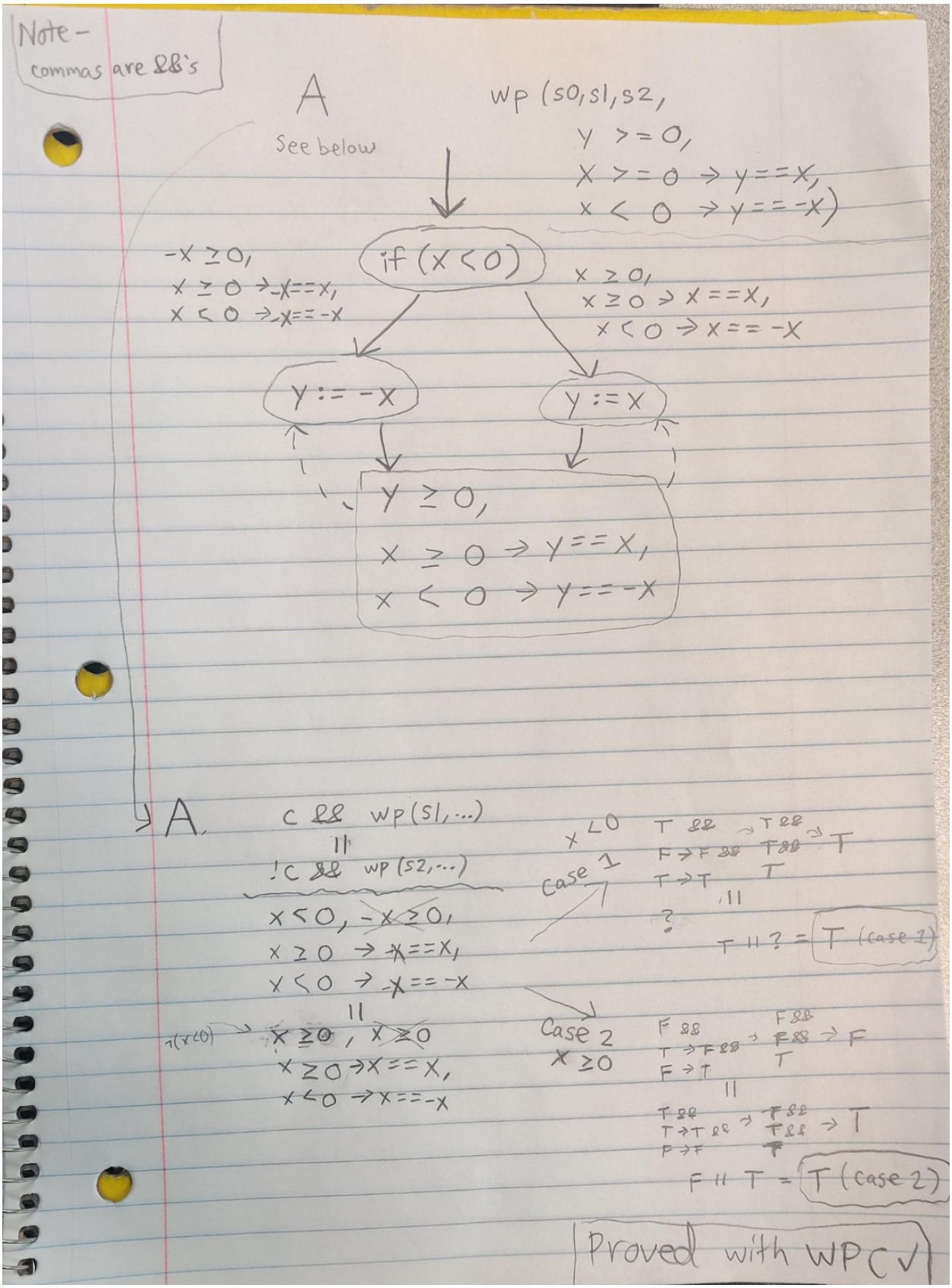
HW6

1. Branching Code Proof

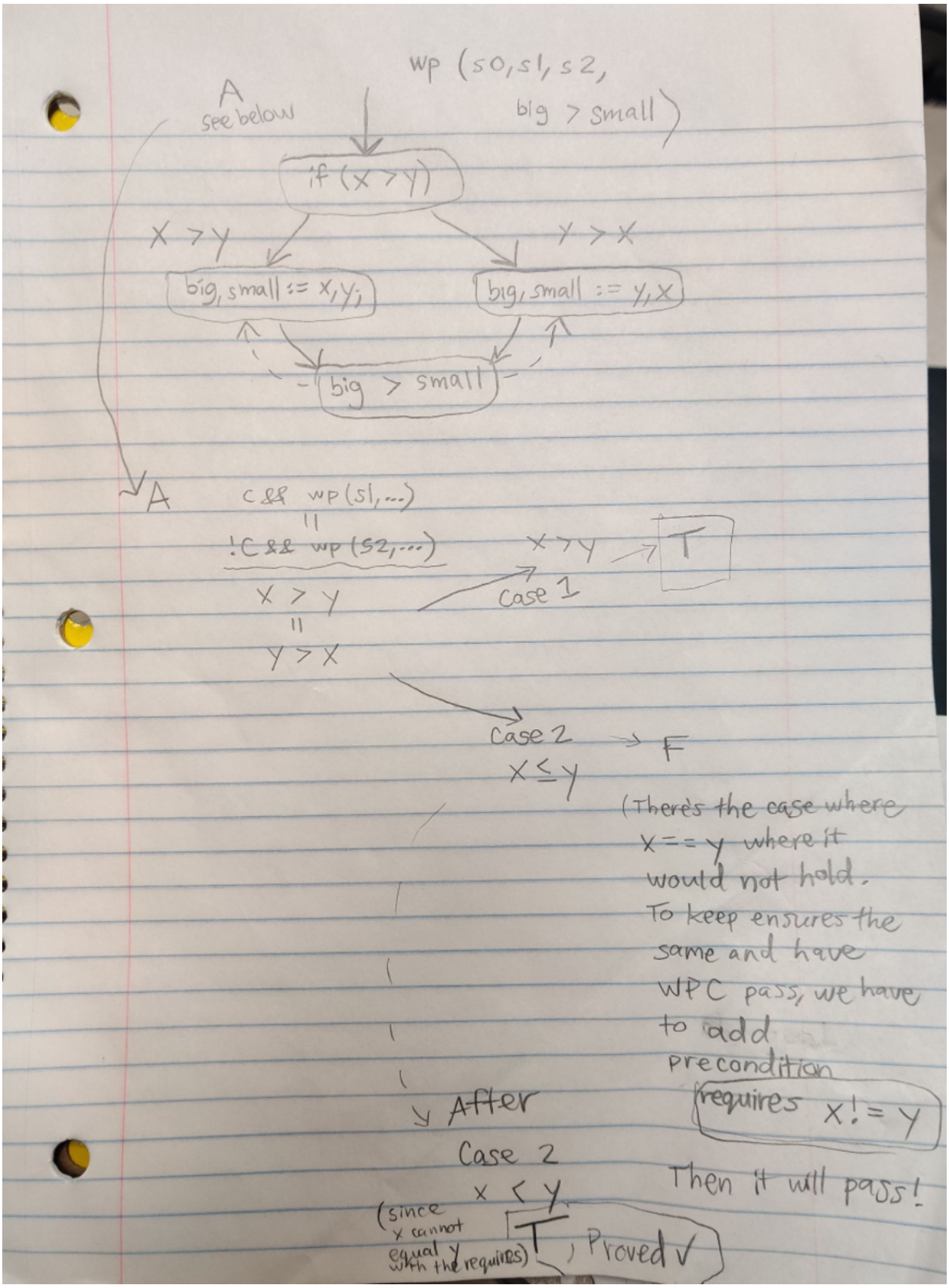
```
method Abs(x : int) returns (y : int)
  ensures 0 <= y;
  ensures 0 <= x ==> y == x;
  ensures x < 0 ==> y == -x;
{
  if (x < 0)
    {y := -x}
  else
    {y := x}
```


Note -
commas are &&'s
A
See below
wp (s0,s1,s2,
y >= 0,
x >= 0 → y == x,
x < 0 → y == -x)
if (x < 0)
-x ≥ 0,
x ≥ 0 → -x == x,
x < 0 → -x == -x
x ≥ 0,
x ≥ 0 → x == x,
x < 0 → x == -x
y := -x
y := x
y ≥ 0,
x ≥ 0 → y == x,
x < 0 → y == -x
A.
c && wp(s1,...)
!c && wp(s2,...)
x < 0, -x ≥ 0,
x ≥ 0 → x == x,
x < 0 → -x == -x
x ≥ 0, x ≥ 0
x ≥ 0 → x == x,
x < 0 → x == -x
Case 1
x < 0
Case 2
x ≥ 0
T && → T && → T
F → F && T && T
T → T T
T || ? = T (case 1)
F && → F && → F
T → F && T
F → T
T && → F && → F
T → T && T
F → F
F || T = T (case 2)
Proved with WP (✓)

See `q1.dfy` for it working in Dafny

2. Complete the Specification

```
method Q2(x : int, y : int) returns (big : int, small : int)
  ensures big > small;
{
  if (x > y)
   {big, small := x, y;}
  else
   {big, small := y, x;}
}
```


wp (s0, s1, s2, big > small)
A
see below
if (x > y)
x > y
y > x
big, small := x, y;
big, small := y, x
big > small
A
c && wp(s1, ...)
||
!c && wp(s2, ...)
x > y
||
y > x
x > y → T
case 1
case 2
x ≤ y
→ F
(There's the case where x == y where it would not hold. To keep ensures the same and have WPC pass, we have to add precondition
requires x != y
Then it will pass!
y After
Case 2
x < y
(since x cannot equal y with the requires)
T, Proved ✓

See `q2.dfy` for it working in Dafny