Prove the correctness of the following “silly” program

```
method M (x0 : int) returns (x : int)
ensures (x0 < 3 ==> x == 1) && (x0 >= 3 ==> x < x0);
{
   x := x0 - 3;
   if (x < 0) {
      x := 1;
    }
  else {
   if (true){
      x := x + 1;
     }
   else {
     x := 10;
    }
  }
}
```

Prove m1 correct!

```
wp(x := e  , R) = R[x →e]
wp(S1 ; S2 , R) = wp(S1, wp(S2,R))
wp(assert B, R) = B && R
wp(if B {S1} else {S2}, R) =
    ( B ==> wp(S1,R)) &&
    (!B ==> wp(S2,R))
```

```
wp(while B I D S, R) =
   I
   && (B && I ==> wp(S,I))
   && (!B && I ==> R)

   && (I ==> D >= 0)
   && (B && I ==>
       wp(tmp := D ; S, tmp > D))
```

```
method m1(n : nat) returns (i : nat)
requires n >= 0
ensures i == 2*n

{
i := 0;
while (i < n)
 invariant i <= n
 variant n-i
 { i := i + 1; }
i := 2*i;
}
```

Prove fib correct!

```
wp(x := e  , R) = R[x →e]
wp(S1 ; S2 , R) = wp(S1, wp(S2,R))
wp(assert B, R) = B && R
wp(if B {S1} else {S2}, R) =
    ( B ==> wp(S1,R)) &&
    (!B ==> wp(S2,R))
```

```
wp(while B I D S, R) =
   I
   && (B && I ==> wp(S,I))
   && (!B && I ==> R)

   && (I ==> D >= 0)
   && (B && I ==>
       wp(tmp := D ; S, tmp > D))
```

```
function fib(n : nat) : nat
{ if n <= 1 then n else fib(n-1) + fib(n - 2) }

method fibFast(n : nat) returns (c : nat)
requires n >= 1
ensures c == fib(n)
{
  var p := 0;
  c   := 1;
  var i := 1;
  while i < n
  invariant 1 <= i <= n
  invariant p == fib(i - 1) && c == fib(i)
  decreases (n - i)
  { var new := p + c;
    p := c;
    c := new;
    i := i + 1;
  }
}
```