Remember these are the steps you must follow when proving an iterative program is correct:

- Clearly state the loop invariant “p” that you will use.
- Use induction to prove that p is a loop invariant. (So for the base step you prove that p is true just before the loop is first entered, and for the inductive step you prove \((p \land cond)\{M_L\}p\) where \(M_L\) is the loop body and cond is the condition.)
- Now use your loop invariant to prove that the given program is partially correct with respect to the given initial and final assertions.
- Finally, prove the given program is correct with respect to the given initial and final assertions (i.e. show that it always terminate).

**Practice Exercises**

1/2. Use induction to prove the partial correctness of the following recursive program to compute \(a^n\) for the initial assertion that \(n\) is a non-negative integer, and the final assertion that the value returned from \(exp(a, n)\) is \(a^n\). Then complete the correctness proof by arguing that this procedure will always halt.

```plaintext
procedure exp(a, n)
  if (n==0) then return 1
  else return a * exp(a, n - 1)
```

3/4. Use induction to prove the following program to compute \(x * y\) is correct given the initial assertion that \(x\) is a non-negative integer, and the final assertion that the value returned from \(mult(x, y)\) is \(x * y\).

```plaintext
procedure mult(x, y)
  if (x==0) then return 0
  else return y + mult(x - 1, y)
```

5/6. Use a loop invariant to prove that the following program is correct with respect to the initial assertion that \(n\) is a positive integer and the final assertion that \(ans = a^n\). (Recall that, by definition, \(a^0 = 1\).)

```plaintext
procedure exp(a, n)
  ans = 1
  i = 1
  while (i ≤ n)
    ans = ans * a
    i = i + 1
  return ans
```
Problems to Submit

1. (15 pts) Use induction to prove the correctness of the following program to compute $a^n$ given the initial assertion that $n$ is a non-negative integer.

```plaintext
procedure fastExp(a, n)
    if (n == 0) then return 1
    else
        if (n/2 == 0) then
            z = fastExp(a, n/2)
            return z * z
        else return a * fastExp(a, n - 1)
```

2. (20 pts) The following program is suppose to determine if $n$ is a power of $b$ (i.e. whether $n = b^j$ for some integer $j \geq 0$) under requirement that both $n$ and $b$ are positive integers. If this program is correct then prove it. If it is not correct, modify the code so that it is correct and then (using a loop invariant) prove that it is correct.

```plaintext
procedure perfectPower(n, b)
    j = 0
    down = n
    while (down%b == 0)
        down = down/b
        j = j + 1
    if (down == 1) then return true
    else return false
```

Challenge Problem

Use a loop invariant to prove that the following program is correct with respect to the initial assertion that $num$ and $\epsilon$ are positive reals and the final assertion that

$$\sqrt{num - \epsilon} \leq approx \leq \sqrt{num}$$

```plaintext
procedure sqrtApprox(\epsilon, num)
    approx = 0
    rem = num
    while (rem > \epsilon)
        d = largeD(rem, approx)
        rem = rem - (2 * approx + d) * d
        approx = approx + d
    return approx

procedure largeD(rem, approx)
    d = 1
    while (rem \geq (2 * approx + d) * d)
        d = 2 * d
    while (rem < (2 * approx + d) * d)
        d = d/2
    return d
```