1.

a. Find a suitable loop invariant

$y \geq 0 \,\&\&\, result * x^y == m^n$

b. Show that the invariant holds before the loop (base case).

Precondition of function: $n \geq 0$

$$n \geq 0, y = n \rightarrow y \geq 0$$

$$result = 1, x = m, y = n \rightarrow result * x^y == 1 * m^n == m^n$$

Thus, invariant $(y \geq 0 \,\&\&\, result * x^y == m^n)$ holds before loop.

c. Show by induction that if the invariant holds after k-th iteration, and execution takes a k+1-st iteration, the invariant still holds (inductive step).

$k$:

$y > 0$ (or we have exited loop) $\&\&\, result * x^y == m^n$

$k + 1$:

Case 1: $y = even\ (y\%2 == 0)$:

$$result_{new} = result$$

$$y_{new} = \frac{y}{2}$$

$$x_{new} = x^2$$

$$result_{new} * x_{new}^{y_{new}} = result * x^{2*\frac{y}{2}} = result * x^y == m^n$$

$$y > 0 \text{ (or we have exited loop) } \&\&\, y_{new} = \frac{y}{2} \rightarrow y_{new} \geq 0\ (y = integer)$$

Case 2: $y = odd\ (y\%2 == 1)$:

$$result_{new} = result * x$$

$$y_{new} = y - 1$$

$$x_{new} = x$$

$$result_{new} * x_{new}^{y_{new}} = (result * x) * x^{y-1} = result * x^{1+y-1} = result * x^y == m^n$$

$$y > 0 \text{ (or we have exited loop) } \&\&\, y_{new} = y - 1 \rightarrow y_{new} \geq 0\ (y = integer)$$

→ Invariant holds for all cases of the $k + 1$ iteration

d. Show that the loop exit condition and the loop invariant imply the postcondition result = m^n.

$$!(y != 0) \,\&\&\, y \geq 0 \rightarrow y == 0 \,\&\&\, y \geq 0 \rightarrow y == 0$$

$$result * x^y = result * x^0 = result$$

$$m^n == result * x^y == result \rightarrow result == m^n$$

e. Find a suitable decrementing function. Show that the function decreases at each iteration and that when it reaches the minimum the loop is exited.

$$D = y$$

$$y = even \rightarrow y_{new} = \frac{y}{2} < y$$

$$y = odd \rightarrow y_{new} = y - 1 < y$$

$y$ decreases at each iteration $\rightarrow$ $D$ decreases at each iteration

$$D_{minimum} = 0$$

loop exits when $y == 0$ && $D = y \rightarrow D_{minimum} = 0$ (loop exits when $D == 0$)

2.

a. Given an array arr[0..N-1] where each of the elements can be classified as red or blue, write pseudocode to rearrange the elements of arr so that all occurrences of blue come after all occurrences of red and the variable k indicates the boundary between the regions. That is, all arr[0..k-1] elements will be red and elements arr[k..N-1] will be blue. You might need to define method swap(arr, i, j) which swaps the ith and jth elements of arr.

```
swap (arr, a, b):
        temp = arr[b]
        arr[b] = arr[a]
        arr[a] = temp

dutch (arr):
        j = 0, k = arr.length
        while j < k:
                if arr[j] == ‘r’ && arr[k-1] == ‘b’
                        swap(arr, j, k-1)

                if arr[j] == ‘r’
                        j++
                else if arr[k-1] == ‘b’
                        k--
```

b. Write an expression for the postcondition.

$$0 <= k <= arr.Length$$
$$AND$$
$$(array\ has\ 'b'\ only\ \&\&\ k == 0\ OR\ arr[i] == 'r', where\ 0 <= i < k)$$
$$AND$$
$$(array\ has\ 'r'\ only\ \&\&\ k == arr.Length\ OR\ arr[i] == 'b', where\ k <= i < arr.Length)$$

c. Write a suitable loop invariant for all loops in your pseudocode.

$$0 \leq j \leq k \leq arr.Length$$
$$AND$$
$$(array\ has\ 'b'\ only\ \&\&\ j == 0\ OR\ arr[i] == 'r', where\ 0 <= i < j)$$
$$AND$$
$$(array\ has\ 'r'\ only\ \&\&\ k == arr.Length\ OR\ arr[i] == 'b', where\ k <= i < arr.Length)$$

3. Fill in the annotations at the designated places. You can use function Factorial in annotations. Fill in the two loop invariants and the assertion.

```
method LoopyFactorial(n: int) returns (u: int)
  requires n >= 0
  ensures u == Factorial(n)
  {
    u := 1;
    var r := 0;
    while (r < n)
      invariant u == Factorial(r) && r <= n
    {
      var v := u;
      var s := 1;
      while (s<=r)
        invariant u == s*v && s <= r+1
        decreases r - s
      {
        u:=u+v;
        s:=s+1;
      }
      r:=r+1;
      assert (u == r*v);
    }
  }
```

Inner loop proof (i = arbitrary iteration, f = next iteration):

I. Base case

$s = 1, r \geq 0, r + 1 \geq 1 \rightarrow s \leq r + 1\ [TRUE]$
$u = v = 1 * v == s * v\ [TRUE]$

II. Induction

Assume $s_i \leq r + 1$ && $u_i == s_i * v$ for arbitrary iteration. For next iteration:
$if\ s_i < r$:
$s_f = s_i + 1,\ u_f = u_i + v$
$s_f \leq r + 1 \rightarrow s_i + 1 \leq r + 1 \rightarrow s_i \leq r$
$s_i < r \rightarrow s_i \leq r + 1\ [TRUE]$
$u_f == s_f * v \rightarrow u_i + v == (s_i + 1) * v \rightarrow u_i == s_i * v\ [TRUE]$
$if\ s_i == r$: Loop exits, assumption still holds.
$s_i == r \rightarrow s_i \leq r + 1\ [TRUE]$
$u_i == s_i * v\ [TRUE]$

III. Imply postcondition

$!(s \leq r) \;\&\&\; s \leq r + 1 \;\&\&\; u == s * v$

$!(s \leq r) \rightarrow s > r$
$s > r \;\&\&\; s \leq r + 1 \rightarrow s == r + 1$
$u == s * v \rightarrow u == (r + 1) * v$ (postcondition before $r = r + 1$)

After exit, $r = r + 1 \rightarrow u == r * v \rightarrow$ matches indicated assert statement

Outer loop proof (i = arbitrary iteration, f = next iteration):

I. Base case

$r = 0, n \geq 0, r \leq n$ [TRUE]
$u = 1 = 0! == r! == Factorial(r)$ [TRUE]

II. Induction

Assume $r_i \leq n \;\&\&\; u_i == Factorial(r_i)$ for arbitrary iteration. For next iteration:
$if\ r_i < n - 1$:
$r_f = r_i + 1 \rightarrow u_f = r_f * u_i = (r_i + 1) * u_i \rightarrow r_f = r_i + 1$
$u_f = Factorial(r_f) = Factorial(r_i + 1) = (r_i + 1)! = (r_i + 1) * r_i!$
$\rightarrow (r_i + 1) * r_i! = u_f = r_f * u_i = (r_i + 1) * u_i$
$\rightarrow (r_i + 1) * r_i! == (r_i + 1) * u_i \rightarrow r_i! == u_i == Factorial(r_i)$ [TRUE]
$if\ r_i == n - 1$: Loop exits, assumption still holds.
$r_i \leq n$ [TRUE]
$u_i == Factorial(r_i)$ [TRUE]

III. Imply postcondition

$!(r < n) \;\&\&\; r \leq n \;\&\&\; u == Factorial(r)$,

$!(r < n) \rightarrow r \geq n$
$r \geq n \;\&\&\; r \leq n \rightarrow r == n \rightarrow u == Factorial(n)$.