Consider the algorithms of MERGESORT and QUICKSORT.

QUICKSORT(A)

if $|A| > 1$ then
    pivot $\leftarrow A[i]$
    partition A into multisets
    $L = \{\text{elements in } A < \text{ pivot}\}$
    $E = \{\text{elements in } A = \text{ pivot}\}$
    $G = \{\text{elements in } A > \text{ pivot}\}$
    QUICKSORT(L)
    QUICKSORT(G)
    $A \leftarrow L, E, G$
fi

Let $G(n)$ ="For all arrays A with n elements from a totally ordered domain, if QUICKSORT(A) is performed, ... unchanged."

We proceed by induction.

Let $n \in \mathbb{N}$ be arbitrary.

Let A be an arbirtrary array with n elements from a totally ordered domain.

Base Case

If $n = 0$ or $n = 1$, the test on line 1 fails and thus A is unchanged and vacuously sorted.

Inductive Step

Suppose $G(n')$ is true for all $n' \in \mathbb{N}$ with $n' < n$.

The test on line 1 succeeds.

By partitioning, all elements in L are less than all elements in E, which are less than all elements in G. A multiset of elements in A is the union of the multiset of elements in L, E, G.

$A[i] \in E$, so $|L|$ and $|G|$ are less than $|A|$.

By the IH, after QUICKSORT(L) and QUICKSORT(G) are performed L and G are sorted in nondecreasing order and the multiset of elements in L and G are unchanged.

After the assignment on line 6, A is sorted in a non-decreasing order. All elements in L are therefore less than all elements in E, which in turn are less than all elements in G, and thu multiset of elements in A is unchanged. By generalisation, $P(n)$.

By induction, for any $n \in \mathbb{N}.P(n)$.

1 Divide and Conquer Algorithms

- divide the problem into smaller parts, often of roughly equal size
- solve each part independently
- combine the solutions for the parts into a solution for the whole problem

1.1 Correctness of Iterative Algorithms

```
z ← 0
w ← y
while w ≠ 0 do
        z ← z + x
        w ← w − 1
od
```

What are the values of the variables immediately after iteration i of the loop? ($w = y - i$, $z = ix$)

Let $P(i)$ ="if the loop is executed at least i times, then immediately afer the iteration i we have $w = y - i$ and $z = ix$".

Note that, by convention, when we talk about the 0th iteration, we are talking about the state immediately before the 1st iteration.

Lemma 1.1

Let $x, y \in \mathbb{Z}$. For all $i \in \mathbb{N}$, we have $P(i)$.

Proof:

Let W_i and z_i denote the values of w and z immediately after iteration i.

Base Case

$w_0 = y = y - 0$ by line 2.

$z_0 = 0 = 0 \cdot x$ by line 1,

so $P(0)$ is true.

Inductive Hypothesis

Let $i \geq 0$ and assume $P(i)$ is true.

Then $w_i = y - i$ and $z_i = ix$.

From lines 4 and 5, we have that $z_{i+1} = z_i + x$ and $w_{i+1} = w_i - 1$, which by inductive hypothesis means that $z_{i+1} = (i+1)x$ and $w_{i+1} = y - (i+1)$, and hence $P(i+1)$ holds.

By induction, we have that $\forall i \in \mathbb{N}.P(i)$.

Corollary 1.2

If the algorithm runs and halts, then at the end $z = xy$.

Proof:

Suppose the loop halts immediately after the iteration i.

From the termination condition of the loop on line 3 we have that $W_i = 0$. By Lemma 1.1,$w_i = y - i$ and $z_i = ix$, so $i = y$ and $z_i = xy$.

A **loop invariant** is a predicate that is true each time a particular place in the loop is reached (often the beginning or end).

Lemma 1.3

$z = x(y - w)$ is a loop invariant which is true at the beginning and end of every iterations.

Proof:

Initially, yrom lines 1 and 2 we can see that $z = 0$ and $w = y$, so $x(y - w) = 0 = z$.

Consider an arbirtrary iteration of the loop.

Let w' and z' denote the values of w and z before the iterations. Let w'' and z'' denote their variable at the end of the iteration.

Suppose the claim holds at the begining of the iteration so $z' = x(y - w')$.

From lines 4 and 5,

$w'' = w' - 1$ and $z'' = z' + x$, so $x(y - w'') = x(y - (w' - 1)) = x(y - w') + x = z' + x = z''$, so the claim is true at the end of the iterations.

By induction, $z = x(y - 0)$ is true after every iteration.