1 Correctness of Algorithms

An algorithm is said to be **correct** if it satisfies its specifications.

A **precondition** is a statement which involves variables used in the algorithm, specifying certain facts must hold before the execution begins.

A **postcondition** is a statement about the variables used in the algorithm, specifying the conditions which must be satisfied when the execution ends without error. Postconditions often identify the correct output for a given input.

Termination is the final state of the halting algorithm, provided the preconditions are satisfied.

If the preconditions hold and the algorithm halts if executed, while also implying that the postconditions hold, the algorithm is said to be **partially correct**.

If an algorithm is partically correct and there is a termination state, then the algorithm is **totally correct**.

Problem. Search an array A for a key k.

Preconditions

- A is finite
- k is of the same type as elements of A or k can be compared to the elements of A

Postconditions

- return true if $k \in A$, false otherwise
- return an integer i such that $1 \leq i \leq \text{length}A$ and $A[i] = k$, or -1 if $k \notin A$
- return a list of all locations in A such that $A[i] = k$
- A and k are not changed by the programme

We alswant to ensure that a programme acesses only elements of A where A is defined.

Consider now an algorithm of binary search

As our preconditions, we have that elements of A and k must belong to a totally ordered set, and A must be sorted in the nondecreasing order.

For the postconditions, we must ensure that elements in the array are permutations of the elements originally in the array, in the nondecreasing order.

Suppose now we want to merge two arrays, U and V, into W.

Preconditions

U and V are sorted in the nondecreasing order.

Postconditions

W is sorted in the nondecreasing and the multiset of elements in W consists of the multiset union of the multiset of elements in U and the multiset of elements in V. We must also make sure that U and V are not changed.

Consider now MERGESORT.

For $n \in \mathbb{N}$, let $P(n)$ ="for all arrays $A[1, n]$ with elements from a totally ordered domain, if MERGESORT(A, n) is performed, then it eventually halts, while A is sorted in the nondecreasing order and the multiset of elements in A is unchanged".

We prove that $\forall n \in \mathbb{N}.P(n)$ by induction.

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

Consider MERGESORT(A, n) for any array $A[1, \ldots, n]$.

Base Case

If $n = 0$, the test on line 1 fails and $A$is unchanged. Vacuously,A is sorted in the nondecreasing order.

Inductive Step

If $n > 1$, suppose that $P(n')$ is true for all $n' \in \mathbb{N}$ such that $n' < n$.

The test on the line 1 succeeds.

By the Inductive Hypothesis, after lines 3 and 4 A' and A'' are sorted in the nondecreasing order and the multiset of elements in A' and A'' are unchanged.

By the correctness of MERGE, the elements of A are in nondecreasing order and the multiset of A, the union of multiset of A' and A', which has the same contents of A.

Conclusion

Hence, $P(n)$ and thus $\forall n \in \mathbb{N}.P(n)$ by induction.