Fully Documented Version of the Algorithm `arrayMax`

A (reasonably) fully documented version of the algorithm `arrayMax` — including the precondition and postcondition for the problem being solved, loop invariant and bound function for the `while` loop in this algorithm, and various other assertions that help to document a proof of the correctness of this algorithm, is as follows.

```
// Precondition:   An integer array A, with some positive length n, is given as input.
// Postcondition:  The largest element in the set
//                                  {A[0], A[1], ..., A[n − 1]}
//                 is returned as output.
integer arrayMax ( integer[] A ) {
    // Assertion: A is an input integer array with some positive length n.
1.  if (A.length == 1) {
      // Assertion: A is an input integer array with length n = 1.
2.    return A[0]
      // Assertion:
      // 1.  A is an input integer array with length n = 1.
      // 2.  The largest element in the set
      //                  {A[0], A[1], ..., A[n − 1] = {A[0]}
      //      has been returned as output.
    } else {
      // Assertion: A is an integer array with length n ≥ 2.
3.    integer i := 0
4.    integer maxSoFar := A[0]
        // Loop Invariant:
        // 1.  A is an input integer array with some length n such that n ≥ 2.
        // 2.  i is an integer variable such that 0 ≤ i ≤ n − 1.
        // 3.  maxSoFar is an integer variable such that
        //                  maxSoFar = max(A[0], A[1], ..., A[i]).
        // Bound Function: n − i, that is, A.length − i
```

```
    // Assertion:
    // 1.  The loop invariant for this while loop is satisfied.
    // 2.  i = 0.
5.  while (i < A.length − 1) {
      // Assertion:
      // 1.  The loop invariant for this while loop is satisfied.
      // 2.  0 ≤ i ≤ A.length − 2
6.    i := i + 1
7.    if (maxSoFar < A[i]) {
8.      maxSoFar := A[i]
      }
      // Assertion: The loop invariant for this while loop is satisfied.
    }
    // Assertion:
    // 1.  The loop invariant for this while loop is satisfied.
    // 2.  i = A.length − 1
9.  return maxSoFar
    // Assertion:
    // 1.  A is an input integer array with some length n such that n ≥ 2.
    // 2.  The largest element in the set
    //                    {A[0], A[1], ..., A[n − 1]}
    //      has been returned as output.
  }
  // Assertion:
  // 1.  A is an input integer array with some positive length n.
  // 2.  The largest element in the set
  //                    {A[0], A[1], ..., A[n − 1]}
  //      has been returned as output.
}
```