Problem 8-1

Burton J. Rosenberg
University of Miami
February 8, 2001

Partition\(A, p, r\)
// Assert \(p < r\)
\(x := A[p]\)
i := p-1
j := r+1
while TRUE
do
repeat j := j-1
   until \(A[j] \leq x\)
repeat i := i+1
   until \(A[i] \geq x\)
if \(i < j\)
   then exchange \(A[i]\) and \(A[j]\)
else return \(j\)

Loop Invariant

\[ i < j, \]
\[ \forall k, p \leq k \leq i, A_k \leq x \]
\[ \forall k, j \leq k \leq r, A_k \geq x \]

The loop invariant is vacuously true before the while loop is run, since \(i < p\) and \(r < j\). We take as a special case when the while loop is run but once. Thereafter we have additionally, \(p \leq i\) and \(j \leq r\), that is, we have non-empty partitions.

Since \(x\) is in the array, the first time through the indices \(i, j\) do not exceed the array. Thereafter, for so long as \(i < j\), the fact that \(A_i \leq x\) prevents \(j\) from decrementing too far. Likewise for \(i\). Hence, except the special case where the loop is run but once, \(p \leq i, j \leq r\), and the array references are safe.

If the while loop is not run only once, then \(j\) is decremented at least twice, which proves \(j < r\) on exit in this case. If the loop is run once, since \(p < r\) on assumption, and \(i == p\) due to choice of \(x\), then \(j == p\). Therefore \(j < r\) in all cases.

Finally, we look at the re-establishment of the loop invariant. If \(i < j\) at the if, we have \(A_i \geq x\) and \(A_j \leq x\). Exchanging, we have \(A_i \leq x\) and \(A_j \geq x\). Since for all \(i' < i\), \(A_i' \leq x\), and likewise for \(j < j'\), the invariant is re-established.
If $i \leq j$ the loop is completed. By the L.I. previously,

\[ \forall k < i, A_k \leq x \]
\[ \forall k > j, A_k \geq x \]

Since the loop is run at least twice, for both statements there are such $k$.

Since $A_j \leq x$ and $j \leq i$ then $A_k \leq x$ for all $k \leq j$. A similar argument shows that $A_k \geq x$ for all $k \geq i$. 