

would need 500 iterations *on average* (the number of pages still to be searched being successively 1000, 999, 998, ...).

3.5 Merging

Suppose that we are given two sorted arrays, $a1$ and $a2$, and are required to make a third sorted array a contain a copy of each value from $a1$ and $a2$. This is the *merging* problem.

We should start by comparing the leftmost (least) component of $a1$ with the leftmost (least) component of $a2$; whichever value is less, we copy that value into the leftmost component of a . We continue in the same way, now ignoring those components of $a1$ and

To merge $a1[left1...right1]$ and $a2[left2...right2]$ into $a3[left3...]$ (where both $a1$ and $a2$ are sorted):

1. Set $i = left1$, set $j = left2$, and set $k = left3$.
2. While $i \leq right1$ and $j \leq right2$, repeat:
 - 2.1. If $a1[i]$ is less than or equal to $a2[j]$:
 - 2.1.1. Copy $a1[i]$ into $a3[k]$.
 - 2.1.2. Increment i and k .
 - 2.2. Otherwise, if $a1[i]$ is greater than or equal to $a2[j]$:
 - 2.2.1. Copy $a2[j]$ into $a3[k]$.
 - 2.2.2. Increment j and k .
3. While $i \leq right1$, repeat:
 - 3.1. Copy $a1[i]$ into $a3[k]$.
 - 3.2. Increment i and k .
4. While $j \leq right2$, repeat:
 - 4.1. Copy $a2[j]$ into $a3[k]$.
 - 4.2. Increment j and k .
5. Terminate.

Algorithm 3.23 Array merging algorithm.

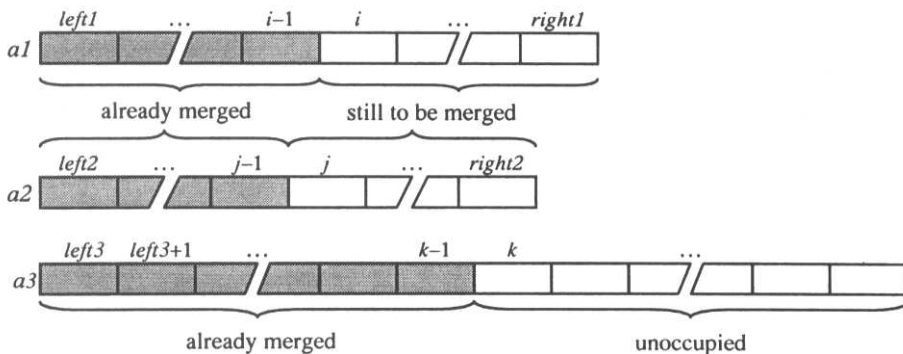


Figure 3.24 Loop invariant for the array merging algorithm.