CS3000: Algorithms & Data — Summer '25 — Laney Strange

Quicksort Algorithm

We went over the divide-and-conquer Quicksort algorithm in class; this handout is so you can see the function typeset in the CLRS pseudocode style. The algorithm partitions the array around a pivot (everything less than or equal on the left and everything larger on the right), and then recursively sorts the remaining subarrays. In the initial call to the QUICKSORT procedure, we give an array A and leftmost index $p = 1$ and rightmost index $r = A.length$.

QUICKSORT(A, p, r)

if $p < r$
    $q =$ PARTITION(A, p, r)
    QUICKSORT$(A, p, q - 1)$
    QUICKSORT$(A, q + 1, r)$

PARTITION(A, p, r)

$x = A[r]$
$i = p - 1$
for $j = p$ **to** $r - 1$
    if $A[j] \leq x$
        $i = i + 1$
        swap $A[i], A[j]$
swap $A[i + 1], A[r]$
return $i + 1$

Here's how we typeset the above functions in CLRS style:

```
\begin{codebox}
\Procname{$\proc{Quicksort}(A, p, r)$}
\li \If $p < r$
\Then
\li $q = \proc{Partition}(A, p, r)$
\li $\proc{Quicksort}(A, p, q-1)$
\li $\proc{Quicksort}(A, q+1, r)$
\end{codebox}

\medskip

\begin{codebox}
\Procname{$\proc{Partition}(A, p, r)$}
\li $x = A[r]$
\li $i = p - 1$
\li \For $j = p$ \To $r-1$
\Do
```

```
\li \If $A[j] \le x$
\Then
\li $i = i + 1$
\li swap $A[i], A[j]$
\End
\End
\li swap $A[i+1], A[r]$
\li \Return $i + 1$
\end{codebox}
```