A variation of Selection Sort that finds the element with the largest value at each step.
Heap Sort takes $O(n \log n)$ steps.

Since $\text{split()}$ is performed $n$ times, where $n$ is the (overall) array length, $O(n \log n)$ steps where $n$ is the subarray length.

In the subarray, $O(\log n)$ steps instead of $O(n)$ steps instead of $O(\log n)$ steps.

- Specifically, $\text{split()}$, which finds the maximum (minimum) element.

Think of Heap Sort as an improved (faster) version of Selection Sort.

• Heap Sort, like Selection Sort, is a hard-split, easy-join method.

Heap Sort
Progress

Unsorted: Heap Sorted

The elements in the unsorted portion of the array are organized into a

heap.

How is split() speed up?
children are heaps.

- the root, if non-null, is the largest key in the tree, and its left and right subtrees are themselves heaps.

exhibits the heap property:

- at most one in path lengths from the root to the leaves (2)

A heap is a binary tree that (1) is almost balanced (we allow a variation

What is a Heap?
Implementing a Heap Within an Array
{'}

return HI;

HeapArena.Singleton.HeapDown(A, 10, 10, HI, HI - 1); // the element at A[10].

// Restore the heap property by "sifting down"

A[10] = temp;
A[HI] = A[HI];
int temp = A[HI];
Swap A[HI] and A[10].

}

public int Split(int[] A, int 10, int HI)

Heap Sort: Split()
The Implementation:

void sittdown(int [] A, int lo, int hi)
\[
A[cur] = \text{dat};
\]

\[
\text{else}
\]

\[
\text{done} = \text{true};
\]

// A[cur] is less than its children.

// heap condition is satisfied.

// A[cur] is less than its children.

// location found for temp.
public class HeapSorter extends Asorter

    Initializing the Heap: HeapSorter()