CSE / ENGR 142 Programming I

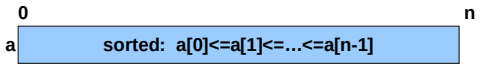
Sorting

Sorting:

- The problem:
  Given an array a[0], a[1], ... a[n-1],
  reorder entries so that
  a[0] <= a[1] <= ... <= a[n-1]
- Many different ways to do it (algorithms)
- Lots of applications
  - faster search (allows binary search)
  - ordering hits in web search engine
  - merging address lists
  - etc.

Sorting Problem

- **What we want: Data sorted in order**

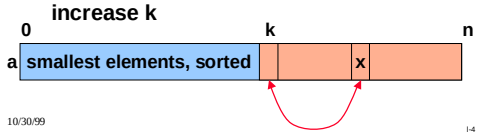


- **Initial conditions**

Selection Sort

- **General situation**

- **Step:**
  - **Find smallest element x in a[k..n-1]**
  - **Swap smallest element with a[k], then increase k**



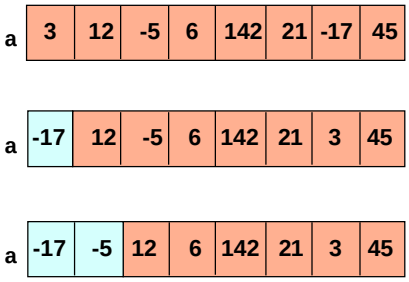
Subproblem: Find Smallest

```
/* Yield location of smallest element in a[k..n-1] */
int min_loc (int a[ ], int k, int n) {
   int j, pos;  /* a[pos] is smallest element */
                /* found so far */
   pos = k;
   for ( j = k + 1; j < n; j = j + 1)
      if (a[j] < a[pos])
         pos = j;
   return pos;
}
```

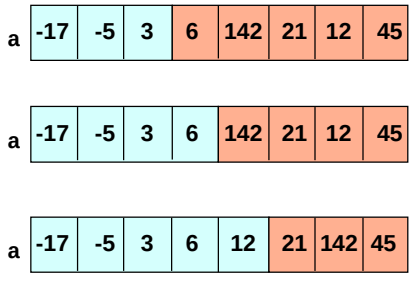

Selection Sort

```
/* Sort a[0..n-1] in non-decreasing order (rearrange
   elements in a so that a[0]<=a[1]<=...<=a[n-1] ) */
int sel_sort (int a[ ], int n) {
   int k, m;
   for (k = 0; k < n - 1; k = k + 1) {
      m = min_loc(a,k,n);
      swap(&a[k], &a[m]);
   }
}
```

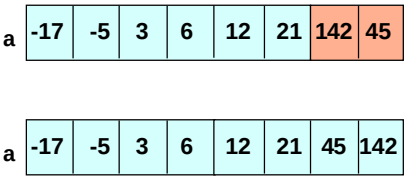
Example



Example (cont)



Example (concl)



Analysis

- How many steps are needed to sort n things?
  - For each swap, we have to search the remaining array; length is proportional to original array length
  - Need *n* search/swap operations
  - Total number of steps proportional to n^2
- Can we do better?
  - Selection, insertion, bubble sorts all proportional to n^2
  - Quicksort (CSE143), heap sort, others, proportional to *n* log *n*, which is much smaller for big jobs (large *n*); best we can do in general case
  - Special cases, can do even better: Sort exams by score: drop each exam in one of 101 piles; work is proportional to *n*