Sorting

- **Sort**: arrange values in a list into some order:
  - Alphabetical
  - Ascending numeric
  - Descending numeric
Sorting

• Selection Sort (in ascending order):
  - Locate smallest element in array. Exchange it with element in position 0
  - Continue until all elements are arranged in order

Selection Sort: example

• Start with an array with the following contents:

  11 2 29 3

• The smallest element is: 2
  Exchange 2 with the element in the first position:

  2 11 29 3
Selection Sort: example

- The next smallest element is 3. Exchange 3 with the element in the second position.

```
2 3 29 11
```

- The next smallest element is: 11 Exchange 11 with the element in the third position:

```
2 3 11 29
```

Selection Sort

- selection sort function (nested loop):

```c
void selectionSort (int list[], int numElems) {
  for (int i=0; i<numElems-1; i++) {
    // find minimum in REST of list (from position i to end)
    // exchange minimum with element at position i
  }
}
```
Selection Sort

• selection sort function (nested loop):

```c
void selectionSort (int list[], int numElems) {
    for (int i=0; i<numElems-1; i++) {
        // find minimum in REST of list (from position i to end)
        int minIndex = i;
        int minValue = list[i];
        for (int j=i+1; j<numElems; j++) {
            if (list[j] < minValue) {
                minValue = list[j];
                minIndex = j;
            }
        }
        // exchange minimum with element at position i
        int temp = list[minIndex]; // put minimum in temp
        list[minIndex] = list[i];  // put i'th elem in minIndex
        list[i] = temp;            // put minimum in i'th position
    }
}
```
Back to the example

• Sort an array of four elements, each iteration
  - when i is 0, j goes from 1 to 3.
    minIndex will be 1, swap list[0] with list[1]
  - when i is 1, j goes from 2 to 3
    minIndex will be 3, swap list[1] with list[3]
  - when i is 2, j goes from 3 to 3
    minIndex will be 3, swap list[2] with list[3]
  - when i is 3, then i < numElems-1 is false
    (3 < 4-1 is false) and we are done.

Sorting Book Inventory

• How would we change the sort function to sort the array of BookEntry?
• Sort using the sku number

    // global
    struct BookEntry {
        int sku;
        string title;
        int quantity;
    };

    // inside main function:
    const int MAX_INVENTORY = 10000;
    BookEntry inventory[MAX_INVENTORY];
Selection Sort: Book Inventory

**sortInventory function:**

```c
void sortInventory (BookEntry list[], int numElems)
{
    for (int i=0; i<numElems-1; i++) {
        // find minimum in REST of list (from position i to end)
        int minIndex = i;
        int minValue = list[i].sku;
        for (int j=i+1; j<numElems; j++) {
            if (list[j].sku < minValue) {
                minValue = list[j].sku;
                minIndex = j;
            }
        }
        // exchange minimum with element at position i
        BookEntry temp = list[minIndex]; // put minimum in temp
        list[minIndex] = list[i];  // put i'th elem in minIndex
        list[i] = temp;            // put minimum in i'th position
    }
}
```

Using sortInventory function

```c
int main {
    const int MAX_INVENTORY = 10000;
    BookEntry inventory[MAX_INVENTORY];
    int numElems = 0;
    getInventory(inventory,numElems); // input inventory (file?)
    sortInventory(inventory,numElems);
    for (int i=0; i<numElems; i++) {
        cout << “sku:” << inventory[i].sku << endl;
        cout << “title:” << inventory[i].title << endl;
        cout << “quantity:” << inventory[i].quantity << endl;
    }
}
```