bubble Sort

- Example bubble sort, ascending order

- 23 7 45 13 64 9 3
- 7 23 45 13 64 9 3 -- swap 23 & 7
- 7 23 13 45 64 9 3 -- swap 45 & 13
- 7 23 13 45 9 64 3 -- swap 64 & 9
- 7 23 13 45 9 3 64 -- swap 64 & 3

- First pass complete

bubble Sort

- Example bubble sort, ascending order

- 7 23 13 45 9 3 64
- 7 13 23 45 9 3 64 -- swap 23 & 13
- 7 13 23 9 45 3 64 -- swap 45 & 9
- 7 13 23 9 3 45 64 -- swap 45 & 3
- Second pass complete

bubble Sort

- Example bubble sort, ascending order

- 7 13 23 9 3 45 64
- 7 13 9 23 3 45 64 -- swap 23 & 9
- 7 13 9 3 23 45 64 -- swap 23 & 3
- Third pass complete

bubble Sort

- Example bubble sort, ascending order

- 7 13 9 3 23 45 64
- 7 9 13 3 23 45 64 -- swap 13 & 9
- 7 9 3 13 23 45 64 -- swap 13 & 3
- Fourth pass complete

bubble Sort

- Example bubble sort, ascending order

- 7 9 3 13 23 45 64
- 7 3 9 13 23 45 64 -- swap 9 & 3
- Fifth pass complete

bubble Sort

- Example bubble sort, ascending order

- 7 3 9 13 23 45 64
- 3 7 9 13 23 45 64 -- swap 7 & 3
- 6th pass
- One more pass to see there is no swap
- So total of 7 passes

Selection Sort – Array based

```
void selectsort( int *a, int len ) {
    int i,j, smallindex, tmp;
    for ( i= 0; i < len; i++ ) {
        smallindex = i;
        for ( j = i+1; j < len; j++ ) {
            if ( a[j] < a[smallindex]) {
                smallindex = j;    }
        }
        if ( smallindex != i ) {  *swap */
            tmp = a[i];
            a[i] = a[smallindex];
            a[smallindex] = tmp; }
    }
```

Selection Sort – Linked list

```
void selectsort( linklist *li) {
    link *current;
    link *current2;
    link *min;
    int tmp;
    current = li->first;
    while ( current != NULL ) {
        min = current;
        current2 = current->next;
        while (current2 != NULL ) {
            if (min->data < current2->data ) {
                min = current2; }
         current2 = current2->next;          }
         tmp = current->data;
        current->data = min->data;
        min->data = tmp;
        current = current->next;     }
}
```

bubble Sort – array

```
void bubblesort( int *a, int len ) {
    int i, tmp;
    bool swapped = true;

    while (swapped == true ) {
        swapped = false;

        for ( i= 0; i <len-1; i++ ){
            if (a[i] < a[i+1]) {
                tmp = a[i];
                a[i] = a[i+1];
                a[i+1] = tmp;
                swapped = true;
            }
        }
    }
}
```

bubble Sort – linked list

```
void bubblesort( linklist *li ) {
    link *current, current2, tmp;
    link *current2;
    bool swap = true;
    while ( swap == true ) {
        swap = false;
        current = li->first;
        current2 = current->next;
        while (current2 != NULL ) {
            if (current->data < current2->data ) {
                tmp = current->data;
                current->data = current2->data;
                current2->data = tmp;
                swap = true;     }
            cout << "st3" << current2->data << endl;
            cout << "min" << min->data << endl;
            current = current2;
            current2 = current2->next;          }
        current = current->next;
    }
}
```