```
1. /*
 2. *
 3. * Implement bubble sort
 4. * Peter Strawn
 5. * CS50 AP
 6. *
7. */
8.
9. #include <cs50.h>
10. #include <stdio.h>
11.
12. #define LENGTH 10
13.
14. int main(void)
15. {
16.
        // declare array of 10 numbers
17.
        int unsorted[LENGTH] = { 9, 7, 3, 4, 2, 8, 1, 6, 0, 5 };
18.
19.
        // print unsorted array
20.
        printf("Unsorted\n");
21.
        for (int i = 0; i < LENGTH; i++)</pre>
22.
23.
            printf("unsorted[%d]: %d\n", i, unsorted[i]);
24.
25.
        printf("\n");
26.
27.
        /* bubble sort */
28.
        // set swaps to any non-zero value
29.
        int swaps = -1;
30.
31.
        // track total number of swaps (just for fun)
32.
        int totalSwaps = 0;
33.
34.
        // repeat until no more swaps (i.e. list is sorted)
35.
        while (swaps != 0)
36.
37.
            // set swaps to zero
38.
            swaps = 0;
39.
40.
            // iterate through numbers in unsorted array
41.
            for (int i = 0; i < LENGTH - 1; i++)</pre>
42.
43.
                // if a number is greater than the number on its right, swap their positions
44.
                if (unsorted[i] > unsorted[i+1])
45.
46.
                    int temp = unsorted[i];
47.
                    unsorted[i] = unsorted[i+1];
48.
                    unsorted[i+1] = temp;
```

```
49.
50.
                    // increment swaps variable for each swap
51.
                    swaps++;
52.
                    totalSwaps++;
53.
54.
55.
56.
57.
        // print sorted array
58.
        printf("Sorted\n");
59.
        for (int i = 0; i < LENGTH; i++)</pre>
60.
            printf("unsorted[%d]: %d\n", i, unsorted[i]);
61.
62.
63.
        printf("\nTotal number of swaps: %d\n", totalSwaps);
64. }
```

```
1. /*
 2. *
 3. * Implement selection sort
 4. * Peter Strawn
 5. * CS50 AP
6. *
7. */
8.
9. #include <cs50.h>
10. #include <stdio.h>
11.
12. #define LENGTH 10
13.
14. int main(void)
15. {
16.
        // declare array of 10 numbers
17.
        int unsorted[LENGTH] = { 9, 7, 3, 4, 2, 8, 1, 6, 0, 5 };
18.
19.
        // print unsorted array
20.
        printf("Unsorted\n");
21.
        for (int i = 0; i < LENGTH; i++)</pre>
22.
23.
            printf("unsorted[%d]: %d\n", i, unsorted[i]);
24.
25.
        printf("\n");
26.
27.
        /* selection sort */
28.
        // track total number of swaps (just for fun)
        int totalSwaps = 0;
29.
30.
31.
        // iterate through numbers in unsorted array
32.
        // stop at second-to-last index value since that will make j the last index value
33.
        for (int i = 0; i < LENGTH - 1; i++)</pre>
34.
35.
            // set index of minimum value to i
36.
            int min = i;
37.
38.
            // iterate through rest of numbers in list to update minimum value of necessary
39.
            for (int j = i + 1; j < LENGTH; j++)
40.
41.
                 if (unsorted[j] < unsorted[min])</pre>
42.
43.
                    min = j;
44.
45.
46.
47.
            // swap positions of i and minimum if i isn't already the minimum
48.
            if (min != i)
```

```
49.
                int temp = unsorted[min];
50.
51.
                unsorted[min] = unsorted[i];
52.
                unsorted[i] = temp;
53.
                totalSwaps++;
54.
55.
56.
57.
        // print sorted array
58.
        printf("Sorted\n");
59.
        for (int i = 0; i < LENGTH; i++)</pre>
60.
61.
            printf("unsorted[%d]: %d\n", i, unsorted[i]);
62.
63.
        printf("\nTotal number of swaps: %d\n", totalSwaps);
64. }
```

```
1. /*
 2. *
 3. * Implement insertion sort
 4. * Peter Strawn
 5. * CS50 AP
 6. *
7. */
8.
9. #include <cs50.h>
10. #include <stdio.h>
11.
12. #define LENGTH 10
13.
14. int main(void)
15. {
16.
        // declare array of 10 numbers
        int unsorted[LENGTH] = { 9, 7, 3, 4, 2, 8, 1, 6, 0, 5 };
17.
18.
19.
        // print unsorted array
20.
        printf("Unsorted\n");
21.
        for (int i = 0; i < LENGTH; i++)
22.
23.
            printf("unsorted[%d]: %d\n", i, unsorted[i]);
24.
25.
        printf("\n");
26.
27.
        /* insertion sort */
28.
        // track total number of swaps (just for fun)
29.
        int totalSwaps = 0;
30.
31.
        // iterate through numbers in the unsorted list (i.e. start at index 1)
32.
        for (int i = 1; i < LENGTH; i++)</pre>
33.
34.
            // set second item in the array as the first unsorted item
35.
            // by default, unsorted[0] is already a sorted array of one item
36.
            int sortValue = unsorted[i];
37.
38.
            // set index of second item in the array as the first unsorted item
39.
            int sortIndex = i;
40.
41.
            // move sortValue down the list so long as two conditions are met:
42.
            // 1. sortIndex does not go below 1 since it's being compared with the number to its left
43.
            // 2. sortValue is less than the value to its left in the array
44.
            while (sortIndex > 0 && unsorted[sortIndex - 1] > sortValue)
45.
46.
                int temp = unsorted[sortIndex - 1];
47.
                 unsorted[sortIndex - 1] = unsorted[sortIndex];
48.
                unsorted[sortIndex] = temp;
```

```
49.
                sortIndex--;
                 totalSwaps++;
50.
51.
52.
53.
        // print sorted array
54.
55.
        printf("Sorted\n");
56.
        for (int i = 0; i < LENGTH; i++)</pre>
57.
58.
            printf("unsorted[%d]: %d\n", i, unsorted[i]);
59.
60.
        printf("\nTotal number of swaps: %d\n", totalSwaps);
61. }
```