**Source File:** `~/2336/32/lab32.cpp`
**Input:** under control of `main` function
**Output:** under control of `main` function
**Value:** 1

The exchange sort is another basic sort algorithm that repeatedly scans a vector and swaps elements until the list is placed in ascending order. The algorithm exchanges a pair of elements when a smaller element is out of order. An illustration of the three-pass process for the four-element list 8, 3, 6, 2 is shown below:

Pass 0: Consider the full list 8, 3, 6, 2. The entry at index 0 is compared with each other entry in the vector at index 1, 2, and 3. For each comparison, if the larger element is at index 0, the two entries are exchanged. After all the comparisons, the smallest element is stored at index 0.

| Initial List | Action | Resulting List |
|---|---|---|
| 8 3 6 2 | Exchange | 3 8 6 2 |
| 3 8 6 2 | No Exchange | 3 8 6 2 |
| 3 8 6 2 | Exchange | 2 8 6 3 |

Pass 1: Consider the sublist 8, 6, 3. With the smallest element already located at index 0, only entries in the vector from index 1 to the end are considered. The entry at index 1 is compared with the other entries at index 2 and 3. For each comparison, if the larger element is at index 1, the two entries are exchanged. After all the comparisons, the smallest element in the new list is stored at index 1.

| Initial List | Action | Resulting List |
|---|---|---|
| 2 8 6 3 | Exchange | 2 6 8 3 |
| 2 6 8 3 | Exchange | 2 3 8 6 |

Pass 2: Consider the sublist 8, 6. With the two smallest elements already located at index 0 and 1, only entries in the list from index 2 to the end are considered. The entry at index 2 is compared with the only other element in the vector, at index 3. After the comparison, the smallest element in the new list is stored at index 2. The resulting vector is ordered.

| Initial List | Action | Resulting List |
|---|---|---|
| 2 3 8 6 | Exchange | 2 3 6 8 |

```
1  #include <cstdlib>
2  #include <iostream>
3  #include <vector>
4  #include <algorithm>
5  #include <chrono>
6  #include <d_random.h>
7
8  using namespace std;
9
10 template <typename T>
11 void exchangeSort(vector<T>& v);
12
13 #include <lab32.cpp>
14
15 const int N = 25000;
16 enum TEST_TYPE {RANDOM, ASCENDING, DESCENDING};
17
18 int main()
19 {
20   vector<int> v, w, x;
21   randomNumber rnd;
22   TEST_TYPE testType;
23   int i;
24
25   for (testType = RANDOM;
26        testType <= DESCENDING;
27        testType = static_cast<TEST_TYPE>(testType + 1))
28   {
29     if (!v.empty())
30       v.clear();
31     switch (testType)
32     {
33       case RANDOM:
34         for (i = 0; i < N; ++i)
35           v.push_back(rnd.random(1000000));
36         cout << "Random Data:" << endl;
37         break;
```

**Figure 1.** `/usr/local/2336/src/lab32main.C` (Part 1 of 2)

Write a function template to implement the exchange sort by using nested for loops. The outer loop has iterations for $pass = 0$ through $pass = v.size() - 2$. The inner loop compares $v[pass]$ with each of the elements at

$$v[pass + 1], v[pass + 2], \ldots, v[v.size() - 1]$$

A sample `main` function for testing your implementation is shown in Figure 1 and a sample execution sequence is shown in Figure 2. You will need to add a target of `lab32main` to the definition of `targets1srcfile` in your `Makefile`.

```
38        case ASCENDING:
39          for (i = 0; i < N; ++i)
40            v.push_back(i);
41          cout << "Ascending Data:" << endl;
42          break;
43        case DESCENDING:
44          for (i = 0; i < N; ++i)
45            v.push_back(N - i);
46          cout << "Descending Data:" << endl;
47          break;
48      }
49      x = v;
50      sort(x.begin(), x.end());
51
52      w = v;
53      auto start = chrono::system_clock::now();
54      exchangeSort(w);
55      auto stop = chrono::system_clock::now();
56      cout << "Exchange Sort: "
57           << chrono::duration_cast<chrono::milliseconds>(stop-start).count()
58           << "ms" << endl;
59      if (x != w)
60        cout << "Sort didn't work correctly" << endl;
61      cout << endl;
62    }
63    return EXIT_SUCCESS;
64  }
```

**Figure 1.** `/usr/local/2336/src/lab32main.C` (Part 2 of 2)

```
newuser@csunix ~> cd 2336
newuser@csunix ~/2336> ./getlab.ksh 32
  * Checking to see if a folder exists for Lab 32. . .No
  * Creating a folder for Lab 32
  * Checking to see if Lab 32 has sample input and output files. . .No
  * Checking to see if /usr/local/2336/src/lab32main.C exists. . .Yes
  * Copying file /usr/local/2336/src/lab32main.C to folder ./32
  * Checking to see if /usr/local/2336/include/lab32.h exists. . .No
  * Copying file /usr/local/2336/src/Makefile to folder ./32
  * Adding a target of lab32main to targets1srcfile
  * Touching file ./32/lab32.cpp
  * Edit file ./32/lab32.cpp in Notepad++
newuser@csunix ~/2336> cd 32
newuser@csunix ~/2336/32> ls
Makefile     lab32.cpp    lab32main.C
newuser@csunix ~/2336/32> make lab32main
g++ -g -Wall -std=c++11 -c lab32main.C -I/usr/local/2336/include -I.
g++ -o lab32main lab32main.o -L/usr/local/2336/lib -lm -lbits
newuser@csunix ~/2336/32> ./lab32main
Random Data:
Exchange Sort: 3258ms

Ascending Data:
Exchange Sort: 909ms

Descending Data:
Exchange Sort: 3472ms

newuser@csunix ~/2336/32>
```

**Figure 2.** Commands to Compile, Link, & Run Lab 32