CS 159 – HW #07
5 Points Possible
Due: April 23, 2012 at 11:00pm.

Problem: While the testing of a program with a random data set is quite common, it is sometimes useful to create special data sets with which one would test a program. For this particular assignment you will create a data set of integers, maximum set size of 25, that is nearly sorted in an ascending order. Replicate the development of the nearly sorted ascending output as seen in the examples that follow.

Example Execution #1:

```
Enter data #1 or -1 to exit: 5
Enter data #2 or -1 to exit: 3
Enter data #3 or -1 to exit: 4
Enter data #4 or -1 to exit: 2
Enter data #5 or -1 to exit: 6
Enter data #6 or -1 to exit: 1
Enter data #7 or -1 to exit: 0
Enter data #8 or -1 to exit: 8
Enter data #9 or -1 to exit: -1

Original Data: 5 3 4 2 6 1 0 8
Final Data: 0 1 2 3 4 5 8 6
```

Example Execution #2:

```
Enter data #1 or -1 to exit: 5
Enter data #2 or -1 to exit: 5
Enter data #3 or -1 to exit: 5
Enter data #4 or -1 to exit: 5
Enter data #5 or -1 to exit: 4
Enter data #6 or -1 to exit: 5
Enter data #7 or -1 to exit: 5
Enter data #8 or -1 to exit: -1

Original Data: 5 5 5 5 4 5 5
Final Data: 5 4 5 5 5 5 5
```

Example Execution #3:

```
Enter data #1 or -1 to exit: 2
Enter data #2 or -1 to exit: 8
Enter data #3 or -1 to exit: 5
Enter data #4 or -1 to exit: 1
Enter data #5 or -1 to exit: 10
Enter data #6 or -1 to exit: 5
Enter data #7 or -1 to exit: 9
Enter data #8 or -1 to exit: 9
Enter data #9 or -1 to exit: 3
Enter data #10 or -1 to exit: 10
Enter data #11 or -1 to exit: 5
Enter data #12 or -1 to exit: 6
Enter data #13 or -1 to exit: 6
Enter data #14 or -1 to exit: 2
Enter data #15 or -1 to exit: 8
Enter data #16 or -1 to exit: 2
Enter data #17 or -1 to exit: 10
Enter data #18 or -1 to exit: -1

Original Data: 2 8 5 1 10 5 9 9 3 10 5 6 6 2 8 2 10
Final Data: 1 2 2 2 3 5 5 5 6 6 8 8 9 10 9 10 10
```

Example Execution #4:

```
Enter data #1 or -1 to exit: 7
Enter data #2 or -1 to exit: 7
Enter data #3 or -1 to exit: 7
Enter data #4 or -1 to exit: 7
Enter data #5 or -1 to exit: -1

Original Data: 7 7 7 7
Final Data: 7 7 7 7
```

Example Execution #5 (no `-1` needed when 25 values [full set] are entered):

```
Enter data #1 or -1 to exit: 10
Enter data #2 or -1 to exit: 9
Enter data #3 or -1 to exit: 8
Enter data #4 or -1 to exit: 7
Enter data #5 or -1 to exit: 6
Enter data #6 or -1 to exit: 5
Enter data #7 or -1 to exit: 4
Enter data #8 or -1 to exit: 3
Enter data #9 or -1 to exit: 2
Enter data #10 or -1 to exit: 1
Enter data #11 or -1 to exit: 10
Enter data #12 or -1 to exit: 9
Enter data #13 or -1 to exit: 8
Enter data #14 or -1 to exit: 7
Enter data #15 or -1 to exit: 6
Enter data #16 or -1 to exit: 5
Enter data #17 or -1 to exit: 4
Enter data #18 or -1 to exit: 3
Enter data #19 or -1 to exit: 2
Enter data #20 or -1 to exit: 1
Enter data #21 or -1 to exit: 10
Enter data #22 or -1 to exit: 9
Enter data #23 or -1 to exit: 8
Enter data #24 or -1 to exit: 10
Enter data #25 or -1 to exit: 9

Original Data: 10 9 8 7 6 5 4 3 2 1 10 9 8 7 6 5 4 3 2 1 10 9 8 10 9
Final Data: 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 8 9 9 9 10 9 10 10 10
```

Example Execution #6:

```
Enter data #1 or -1 to exit: 10
Enter data #2 or -1 to exit: 9
Enter data #3 or -1 to exit: 8
Enter data #4 or -1 to exit: 7
Enter data #5 or -1 to exit: 6
Enter data #6 or -1 to exit: 5
Enter data #7 or -1 to exit: 4
Enter data #8 or -1 to exit: 3
Enter data #9 or -1 to exit: 2
Enter data #10 or -1 to exit: 1
Enter data #11 or -1 to exit: 10
Enter data #12 or -1 to exit: 9
Enter data #13 or -1 to exit: 8
Enter data #14 or -1 to exit: 7
Enter data #15 or -1 to exit: 6
Enter data #16 or -1 to exit: 5
Enter data #17 or -1 to exit: 4
Enter data #18 or -1 to exit: 3
Enter data #19 or -1 to exit: 2
Enter data #20 or -1 to exit: 1
Enter data #21 or -1 to exit: 10
Enter data #22 or -1 to exit: 9
Enter data #23 or -1 to exit: 8
Enter data #24 or -1 to exit: 10
Enter data #25 or -1 to exit: -1

Original Data: 10 9 8 7 6 5 4 3 2 1 10 9 8 7 6 5 4 3 2 1 10 9 8 10
Final Data: 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 8 9 9 10 9 10 10 10
```

Academic Integrity Reminder:

- Please review the official policies and consequences of the course as they relate to academic integrity. The assignment you submit should be your own original work. You should be consulting **ONLY** course staff regarding your specific algorithm for assistance. **Collaboration is not permitted** on individual homework assignments.

Additional Very Important Requirements and Reminders:

- Accept input and produce output formatted **exactly** as seen in the example executions above.
  - There is no input to validate for this assignment. All items in the data set will be non-negative.
  - Do not add any "bonus" features not demonstrated in the example executions provided.
- **DO NOT** use any material found outside of the first EIGHT chapters of the C text.
  - You may make use of any sorting algorithm found in the C programming text or course notes packet. Be sure to list your source in the sorting function header. All code must be comply with course standards.
- Additionally, **this assignment MUST make good use of user-defined functions to be considered for partial credit.**
- A program **MUST** compile to be considered for partial credit. The submission script will reject the submission of any file that does not compile and is therefore untestable.

Course Programming and Documentation Standards Reminders:

- Details regarding the course expectation of user-defined function use and documentation can be found in the standards document and chapter 4 notes.
- Always use { and } with all relevant selection and repetition constructs. Code inside of such constructs should be indented two additional spaces.
- Select meaningful identifiers (names) for all variables (including parameters) and symbolic/defined constants in your program.
  - The maximum size of the data set is known before you compile the program and would be a great candidate for a symbolic/defined constant.
- Indent all code found within the `main` function **exactly** two spaces.
  - All code found within a selection or repetition construct will be indented exactly two additional spaces.

When you submit... only the last attempt of a submission is kept for grading. All other submissions are over-written and cannot be recovered. You may make multiple submissions but only the last attempt is retained and graded.

- Verify in the e-mail sent to you by the course that you have submitted the correct file, to the correct assignment (`hw07`), and to the correct lab section. Forwarding confirmation e-mails from Purdue to external e-mail services may result in the mail being undelivered or end up being identified as spam.
- Leave time prior to the due date to seek assistance should you experience difficulties completing or submitting this assignment.
- All attempts to submit via a method other than through the sage server as set up during the first week of the semester will be denied consideration. Leave time to seek assistance should you struggle to use any of the tools of the course.

Assignment deadlines... are firm and the electronic submission will disable promptly as advertised. We can only grade what you submit as expected prior to the assignment deadline.

All course programming and documentation standards are in effect for this and each assignment this semester. Please review this document in your course notes packet.