S-Q 1997 - Assignment 2

Due 07/07/97 at 6:00pm

Last Date Accepted: 07/10/97 at 6:00pm

1 Getting Prepared

1. Copy over the files for this assignment. Copy all of the files from `~libsq/assts/asst2` into your directory `~/Q/asst2` (which should have been created when you ran `q-setup` for assignment 1).

2 Exercises (100 points)

All work should be done in directory `~/Q/asst2`. The `Makefile` provided includes rules to build all the programs in this assignment; you shouldn't need to change it at all.

1. (35 points) Complete the functions specified in `dllist.c`. Test your functions carefully with `dllist_test`, to make sure that they give the same results as the reference implementation.
2. (15 points) Complete the functions specified in `deque.c` (and described in Weiss, exercise 3.26). Note that `deque.c` uses the doubly linked list structures from dllist; if you have troubles getting your dllist functions to work, feel free to use the dllist_dbg functions instead of your own. Test your functions carefully with `deque_test`, to make sure that they give the same results as the reference implementation.
3. (25 points) Weiss, problem 3.10. Instead of answering questions a, b, and c as written, do the following:
   (a) Devise and implement an algorithm that **you are sure is correct and solves the problem**. *Do not be overly concerned with efficiency.* Implement your algorithm by completing the function `josephus` in `joe1.c`. You can test your program by building and running the `joe1` program, and comparing its output with that of `sol_joe`.
   (b) Make a copy of `joe1.c` named `joe2.c`, and implement any algorithmic optimizations as you can think of in it (preserving correctness, of course). Be sure to adequately comment your code! Test your program by building and running the `joe2` program.
   Do not get carried away on this part of the problem– you could spend days tinkering with it, but remember that this problem is only a small part of the assignment!
   (c) Place your answers to the following questions in file `joe.txt`:
      i. Analyze your joe1 and joe2 algorithms. What are their best-case and worst-case running times, in big-O notation?
      ii. Using the `test-joe` script, compare the speeds of `joe1`, `joe2`, and `sol_joe`. How much faster (or slower) are each compared with the others?
      iii. Does the relative speed of joe1, joe2, and sol_joe appear to depend on some relationship between N and M? If so, describe this relationship. (If you have an exact equation, give that, but otherwise be as specific as you can be.)

iv. Did you need to do anything really complicated to get much speedup? Discuss the tradeoff between speed and simplicity.

v. Briefly describe any optimizations you considered, but didn't implement because they were too difficult.

You will probably find that a linked list is an appropriate data structure to represent the circle of players (but in this case, the linked list will have its ends tied together to create a loop, instead of having a head and tail). Feel free to use the functions you wrote for dllist.c or deque.c (or the debugging versions of these functions), if they are useful.

Be sure that your code **free**'s all of the memory it allocates.

4. (25 points) Implement the **Balancing Symbols** algorithm described on page 71 of Weiss, by writing the `balance_symbols` function (and any other functions you need) in `balance.c`. Refer to `balance.c` for more detail. When complete, the output from your program should be the same as that from `sol_balance`.

   You may implement your own stack routines (based on your deque.c code, perhaps), or use the code provided in `istack.c` and `istack.h` in any way you see fit. You may assume that balanced symbols are never nested very deeply (at most 100 levels), so using a finite stack (with a maximum depth of 100) is sufficient.

3 Submit Your Work

1. Use the `submit` program to hand in your work. From your directory `~/Q`, run the `submit` program. The name of the course is **cscisq**, and the name of the project is **asst2**, and the name of the file to submit is also **asst2**. (See the UNIX tutorial for more info about `submit`.)

2. Print out a copy of your dllist.c, deque.c, joe1.c, joe2.c, joe.txt, and balance.c, and hand them in during the next class or section. (If you modify or create any other files, print them out as well.)