Assignment 2 COMP2111 17s1
`quicksort`

Kai Engelhardt

Revision: 1.7 of Date: 2017/05/04 08:22:30

This assignment is worth 17 marks and due before the end of week 8, that is, **Thursday May 4th, 23:59:59** local time Sydney. Assignments are done in pairs.

Problem Statement

> *Quicksort* (sometimes called *partition-exchange sort*) is an efficient sorting algorithm, serving as a systematic method for placing the elements of an array in order. Developed by Tony Hoare in 1959, with his work published in 1961, it is still a commonly used algorithm for sorting. (Wikipedia)

In this assignment, we're deriving a recursive version of quicksort for arrays of integers.

Tasks

1. Specify a procedure that upon input of an array a of integers, a left bound ℓ, a right bound r, and an array length N, sorts $a[\ell..r]$ in place from smallest to largest by $\leq$ without affecting the remainder of a.

2. Use refinement calculus laws to formally derive a recursive implementation of your specification that uses the quicksort algorithm to sort the sub-array in-place. (There are various choices to be made including how to pick a pivot and how to perform the partitioning step. These are deliberately left unspecified here.)

3. Translate your implementation into a C function with the prototype as provided in `quicksort.h`.

```
/* Sort a[l..r] using quicksort
 *
 * @a array
 * @l left index; 0 <= l < N
 * @r right index; 0 <= r < N
 * @N length of a
 */
void quicksort(int a[], int l, int r, int N);
```

This should go without saying, but do not use the C library function `qsort`.

4. Describe your solutions to tasks 1–3 in a LaTeX document that your tutor enjoys reading. In more detail:

    - Describe how you formally derived the implementation from the specification. List all arising proof obligations (premises to laws) and discharge them by proof. Properly reference any outside sources you use.

    - Discuss and justify the changes made during the translation to C. Describe the limitations of your implementation should there be any.

Testing

I'm providing a simple test harness in `ass2/C`. Once compiled with `make`, the program `quicksorttest` reads in an array from stdin, runs and times both your quicksort code and the C standard library function qsort on the input, checks your result, and prints timing information. The program `genrand` creates input suitable for `quicksorttest`. Some simple test files can be found in `ass2/tests`.

Deliverables

`quicksort.c` C source of your 3 solution.

`quicksort.tex` is a LaTeX document with your names or student numbers mentioned in the `\author` command. It contains your task 4 solution.

Examples

Examples of our interaction with your submission are as follows. (Our shell prompt is $ and user input is coloured red.)

```
$ make
$ cc -O -Wall -Werror -c quicksort.c
$ cc -O -Wall -Werror quicksort.o quicksorttest.c -o quicksorttest
$ cc -O -Wall -Werror    genrand.c   -o genrand
$ ./genrand 1000000 | ./quicksorttest
Timing user code:
cpu  : 0.11 secs
user : 0 secs
Timing qsort for comparison:
cpu  : 0.24 secs
user : 1 secs
$ ./quicksorttest < tests/10000r10x10.txt
Timing user code:
cpu  : 0.06 secs
user : 0 secs
Timing qsort for comparison:
cpu  : 0.08 secs
user : 0 secs
$ make quicksort.pdf
<...>
$
```

Submission Instructions

When submissions are enabled, the `give` command to be run is:

```
    % 2111
    % give cs2111 ass2 quicksort.c quicksort.tex
```

The command above submits the bare minimum. Should you feel the need to include more files just list them as well. Do not submit any of the provided files. The provided subject-specific LaTeX style files and assignment-specific files will be copied into the directory where your submission is built.

Bonus Task

Submissions that succeed in deriving Hoare's original partitioning step attract up to **2 bonus marks**. Hoare's original partitioning step is interesting because it uses substantially fewer comparison operations than most of the simpler versions published later.

```
$Log: ass2.tex,v $
Revision 1.7  2017/05/04 08:22:30  kaie
Summary: removed "Makefile" from submission example

Revision 1.6  2017/04/29 21:14:30  kaie
Summary: Thursday is the 4th not the 5th

Revision 1.5  2017/04/28 10:38:28  kaie
Summary: added '/' to certain URLs

Revision 1.4  2017/04/27 03:53:02  kaie
Summary: deadline pushed back

Revision 1.3  2017/04/16 02:41:34  kaie
Summary: added length of a as a parameter; C doesn't really need it but the derivation does

Revision 1.2  2017/04/13 11:14:59  kaie
Summary: bonus task added

Revision 1.1  2017/04/11 07:30:15  kaie
Initial revision
```