Exercise One : Sorting

The goal of this exercise is to investigate ways to do sorting in parallel, using threads.

Code provided to get you started

We will be sorting 100 byte records taken from files. This idea was taken from the "Sort Benchmark" hosted at http://sortbenchmark.org/ – see that web page for much more information.

There are three tools here:

1. **gensort** – a tool for generating data to be sorted.
2. **valsort** – a tool for validating sorted output.
3. **dumbSort** – a non-parallel (single-threaded), very simple sorting program that takes input created by gensort and creates output which can be validated by valsort. This code was written by Prof. R. C. Moore (ronald.moore@h-da.de) for the P&DC course.

To build the tools, simply enter "make" on the command line.

The first two tools (gensort and valsort) are built in the directory **gensort-1.5**. You should use these unchanged (unless you have *very* good reasons for changing either of them). The original version of this software is available at http://www.ordinal.com/gensort.html. The readme file and the license are also original. Build these tools by simply typing "make" on the command line.

The third tool (dumbSort) is a very simple C++ implementation that uses the standard C++ vector to store records and the standard C++ sort to sort them (and the standard C++ clock functions to report the time spent in input, sorting and output).

Example Usage

To create a binary file called four.dat with 4 records (400 bytes long), call (on the command line)

```
./gensort 400 /tmp/four.dat
```

Note that the file is in the local "tmp" directory, so that we are not cluttering up our home directory. You can check that this file is not sorted (yet) by calling

```
./valsort /tmp/four.dat
```

To create a new file with the same 4 records in *sorted* order, call

```
./dumbSort /tmp/four.dat /tmp/sortedFour.dat
```

You will probably want to validate this file by calling

```
./valsort /tmp/sortedFour.dat
```

When we're done, it is simply polite to delete our data files:

```
rm -iv /tmp/*.dat
```

Exercise 1 – Your Assignment

Your job now is to create a parallel, multi-threaded sort program. You may start with the code from the dumbsort program if you like, or you might want to start from scratch. This is up to you.

Tip: You can find large (read-only) test input on the BDC in the directory

```
/home/datasets/sortbenchmark.org
```

After you have your own program up and running, start measuring its performance and write a lab report, as described in Moodle https://lernen.h-da.de.

Prof. R. C. Moore