Prof. Dr. Alexander Bockmayr,
Prof. Dr. Oliver Serang,
Christopher Pockrandt

October 29, 2015

Algorithms - Programming Exercices

WS 2015/16

Exercise 2
Due date: 12.11.2015 until 11 pm.

Hashing with Open Addressing

1. Implement a class openAddMap that implements a hash table with the following public member functions:

   - `size_t size()` - returns the size (reserved slots) of the table.
   - `size_t numElem()` - returns the number of elements currently in the table (not including `NIL` markers).
   - `bool insert(unsigned int)` - insert a new element into the table. On success return `true`. If element was already in the table return `false`.
   - `bool find(unsigned int)` - search for entry in the table. If found return `true` otherwise `false`.
   - `bool remove(unsigned int)` - delete entry from table. If deleted return `true`. If element was not in the table return `false`.
   - `openAddMap(const std::string & mode)` - Constructor takes a `std::string` argument which is either "linear", "quadratic" or "double" and defines which probing scheme your table should implement.

2. The load factor (excluding NIL slots) should always be between 25% and 75%. You can use `-1u` as marker for deleted slots and `-2u` as marker for free slots.

3. The class shall be defined in a file `openAddMap.h` and the member functions shall be defined in `openAddMap.cpp`.

4. Visualise the distribution of items in your table for the different probing schemes. Can you observe primary/secondary clustering?

5. Evaluate the number of probes you need for the probing schemes and the runtime.

6. There will be a skeleton file exercise2.cpp in the `data` directory, which can be used as minimal sanity check. It requires that the source `openAddMap.cpp` and `openAddMap.h` exist. There will also be an updated `Makefile` containing the base structure to compile exercise2.

7. Together with your program you have to provide a short application note (1-2 DIN-A4 pages; pdf) describing the implementation and an evaluation of the runtimes and discussing the remarks above.

8. The due date is Wednesday 12^{th} of November 2014 until 1 pm at the latest. All versions submitted later than this time stamp won't be assest.

9. The program shall run on a linux pool machine (see the wiki for additional information `http://www.mi.fu-berlin.de/w/ABI/ProgrammingExercisesWS14`).

10. The compiler flags shall be set to `-pedantic -Wall -ansi -O3`.

You can score 4 pts.. 3 pts. for the program and 1 pt. for the application note.
3 pts. = The program compiles and runs successfully with no errors.
2 pts. = The program compiles and contains minor errors.
1 pt. = The program compiles and contains some critical errors.
0 pts. = The program doesn't compile or does not meet the requirements.