COMP20003 Algorithms and Data Structures

Worksheet 7

[week starting 4 of September]

Second (Spring) Semester 2016

Overview

The workshop for Week 7 will start with a tutorial on Mergesort and Quicksort, and you will implement a HashFunction/HashTable or work on your assignment.

Tutorial Questions

Question 6.1 You are asked to show the operation of quicksort on the following keys. For simplicity, use the rightmost element as the partition element:

```
2 3 97 23 15 21 4 23 29 37 5 23
```

Comment on the stability of quicksort and its behavior on almost sorted inputs.

Question 6.2 Trace the action of bottom-up mergesort on the same keys:

```
2 3 97 23 15 21 4 23 29 37 5 23
```

Is mergesort stable?

Programming exercises

Programming 6.1 You can work on your assignment or on the following programming excercise.

Implementing a hash table will help you understand how it works. In this laboratory, you are asked to implement an open addressing hash table that uses linear probing for collision resolution. In the interests of simplicity, and to help you implement the hash table quickly, your table can simply store integers (keys), keep the table very small (suggested size 13, why?), and use a very simple hash function, e.g. `f(key) = (key*97)%13`. Implement a function `printTable(ht)`, so that you can check what is in the hash table at every step. Remember that you will get a segmentation fault if you try to print a `NULL` value, so wrap the printing within a conditional statement that tests for `NULL`.

Because the table is small, you can engineer collisions. Check what happens when you get a collision, then when you get two collisions, etc. Think about how you could handle the situation where the table is almost full – how nearly full are you going to let it get? Could you expand the table? If you expand the table what will happen to the items that are already in the table?

September 2, 2016