Algorithms – 2015/16

# Topics per lesson

|  |  |  |
| --- | --- | --- |
| **Week** | **Topic** | **Book paragraphs** |
| 1 | Complexity of algorithms (empirical analysis, O notation) | 1.4 |
| 2 | Insertion sort; Merge sort; | 2.2  1.3 |
| 3 | [List?], Queue, Stack, Bag; Hash tables | 3.4 |
| 4 | Binary search trees | 3.2 |
| 5 | Balanced search trees: 2-3 search trees (no deletion) | 3.3 |
| 6 | Graphs (undirected, directed, Dijkstra shortest path) | 4.1, 4.2, 4.4 |
| 7 | Dynamic programming (Fibonacci example); Floyd-Warshall | / |
| 8 | Summary of the course |  |

# Assessment

* Combination of **written exam** and **practical assignment**
  + Written exam must be sufficient () to pass
  + Grade: 100% practicum

# Practical assignments

General information:

* 4 programming assignments (A1, A2, A3, A4), each one with its own deadline
* **Oral checks** to verify authorship of code
  + ask to rewrite some small parts of the code
* **Commit history** on Github to enforce the deadlines
  + oral check is based on what was committed *before the deadline*
* **Upload** everything **on** **N@tschool** at the end, to store assignments for accreditation
* Language: **C#**, because the students will receive a supporting visualization framework in which to work. The assignments will be based on the simulation of a city (houses, hospitals, shops, etc. all connected by streets). The students must implement algorithms to answer some queries on the simulated city.

Assignments details:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Assignm.** | **Query to answer** | **Course topic** | **Deadline** | **Points** |
| A1 | Sort services (hospital, shops, …) by Euclidean distance from a specified house | Mergesort | End week 4 | 2 |
| A2 | One at choice between:  a) Find the closest hospital/service from a user-selected house within Euclid. distance  b) Find the closest hospital/service from a user-selected house | a) Hash table  b) Binary search tree | End week 6 | a) 2  b) 3 |
| A3 | Shortest path from a user-selected house to a user-selected hospital/service   * [[Compute also the probability that the patient dies during that path (given that each street is associated to a “danger” coefficient)]] | Dijkstra | End week 8 | 2 |
| A4 | Shortest path from a user-selected house to all hospitals/services | Floyd-Warshall | End week 9 | 2 |

The total points of the assignments is the sum + 1: maximum possible is 10.

Examples:

* A1 and A2.a 🡪 2+2+1 = 5
* A1 and A2.b 🡪 2+3+1 = 6
* A1 and A2.a and A3 🡪 2+2+2+1 = 7
* A1 and A2.a and A3 and A4 🡪 2+2+2+2+1 = 9