Universidad
de Alcalá

Laboratory Assignment 3

Data Structures

Fall 2013

Terms and conditions:

- Assignment should be done in groups (preferred size of 2, max. 3 people).
- Assignment shall be zipped and submitted to BlackBoard. Accepted formats are .zip, .rar and .7z.
- Rename the zipped file as *Surname_Firstname* before submission. Only one person per group will send the assignment and the file will be named after this sender.
- Assignment shall include the source files/projects as well as a document (.doc or .pdf). The document must also include the IDs and names of all the members of the group.

Students have to present every assignment to the teacher and answer the teacher's questions about it on the corresponding lab session (see deadlines).

Submission deadlines:

- Assignment 1: 5-Nov-2013
- Assignment 2: 3-Dec-2013
- Assignment 3: 17-Jan-2014

Submissions that do not meet the deadlines and/or the requirements may not be reviewed. It is responsibility of the students to meet the requirements and it is not responsibility of the teacher to warn students and go after students when such requirements are not met. Exceptions will not be considered (except for those included in the examination regulation-"*normativa de examenes*").

Students that do not pass the practical part of the course (i.e. the lab) on the January call may be required to complete an additional assignment for the extraordinary call (June).

Important note: On these assignments only basic requirements are described. Students need to do additional assumptions and to make design decisions based on the given requirements. Students also need to decide on the input data and on the way in which that data is inputted into the system. Sample data needs to be provided to facilitate testing. Students are free to implement the solution in the way they prefer. Any decision will be reasonable providing that (1) it is explained on the written document and commented on the source code, and (2) it does not contradict blatantly the requirements stated here.

Assignment 3: Trees – *Palindrome problem*

Results from assignment 1 should be used in this assignment.

The problem is described as follows:

The program will read words from an input source (standard input or text file[1]) and store words in a search tree (all words, palindromes and not palindromes). The tree must be sorted using alphabetical criterion and must remain sorted after every new insertion.

Each node of the tree must contain one word and one attribute to indicate whether or not that word is palindrome (using assignment 1 for checking this).

The operations needed, at least, are the next:

- **Add** a new word to the search tree (keeping the tree sorted).
- **Search** and show on the screen those words that are repeated in the search tree (one word is repeated if it appears at least two times in the tree. The number of repetitions is not relevant). For each word you have to show whether or not is palindrome.
- **Traverse** the search tree and show all words that are palindromes. These words have to be showed alphabetically sorted.
- **Remove** all words that are not palindromes keeping the structure of the search tree.

When the assignment is completed, the documentation should contain the sections of the following index:

Index

1. Implementation details
    - ADT specifications: Class name that contains the ADT
    - Types defined in the unit
    - ADT Operations definition (name, arguments and return)
    - Design of the relationship between the ADTs implemented, if any.
    - Explanation of the highlight methods on the ADTs or their relationship
2. Solution adopted to implement specifications

[1] On keyboard input, words must be read one by one. On file input, every line must contain only one word.

3. Organization charts
    a. UML diagram or similar to get appropriate representation of every element designed
4. Execution time of the program
5. References (Bibliography)