Carles Toujouse i Astor Prieto

Practica 2 – Sistemes Operatius II

- ## Comentaris de la Implementacio:

En aquesta segona practica, es demanava que usant el codi base donat al completar la practica 1, modifiquem el comportament del programa per a que ara les linies del fitxer de dades es llegeixin en blocs de ‘N’ linies, es processen i insereixen previament a una taula Hash i finalment es bolquen les dades de la taula al arbre binari. Aquesta practica es la implementacio previa que s’ha de fer per a poder, en futures practiques, tenir un bon programa que aprofiti el *multithread* dels ordinadors.
La implementacio es basa en tres grans funcions que implementen respectivament les noves funcionalitats mencionades: ‘read_lines()’, ‘save_to_hash()’ i ‘save_to_tree()’.

  - ### Lectura de ‘N’ linies del fitxer ‘csv’:

D’aquesta funcionalitat s’en encarrega ‘*read_lines()*’. Aquesta funcio usa el parametre definit al **main** ‘N’ per a saber el bloc de linies a llegir del fitxer de dades i usa un *array* de *strings* per a guardar-les per a la posterior insercio a la taula Hash. La implementacio de la funcio es principalment reaprofitada del codi de la practica 1, on es llegien totes les linies del fitxer, una per una.

  - ### Insercio de dades a la taula Hash:

La funcio encarregada de inserir les dades a la taula hash, ‘*save_to_hash()*’, pren les linies guardades per la funcio ‘*read_lines()*’ i les insereix a una llista de *structs* ‘*List*’, donat a la implementacio de *linked-list.h*, que la usarem com a taula hash. La taula es indexada amb el hash obtingut del aeroport de origen usant la funcio auxiliar ‘*get_hash()*’ i emmagatzema tota la informacio que serà necessaria en la insercio al arbre binari. La principal diferencia en la implementacio en aquesta part respecte a la practica primera es troba en la funcio de llibreria de C que utilitzem per a obtenir els *tokens* de cada linia. A la primera usavem ‘strtok()’, que podia donar fallos al processar les linies al no tenir en compte els espais buits com a valors. En aquesta usem ‘strsep()’, que te en compte totes les divisions generades per els separadors definits.

- ## Bolcat de la taula Hash al Arbre Binari:

La funcio que s'encarrega de bolcar la taula hash al arbre binari es '*save_to_tree()'*, que funciona de la manera següent, primer de tot busca si el valor existeix, de ser així actualitza els valors dels retrasos dels vols, i si no es troba el valor, el creem, tant el node com la seva llista.

En aquet punt del codi tenim un problema que no hem conseguit solucionar, a la hora de buscar i inserir els nodes al arbre. En buscar els nodes al arbre, el programa troba els nodes tot i que encara no s'han arribat a inserir. En el nostre cas, insereix el primer node correctament i troba el seu valor a la següent iteracio; a partir d'aqui sempre troba els seguents nodes i els va actualitzant com si ja existisin. En un principi vam pensar que l'error podia provindre al fer una mala copia de les dades de la taula hash per a l'arbre, però tot i copiar-les en memoria i no sobreescriure-les, el error de cerca al arbre persisteix, i els nodes no inserits al arbre els continua trobant amb les seves dades correctes.