2018-19

Sistemes Operatius II

PRÀCTICA 3

FRANCISCO DÍAZ RUIZ NIUB: 16828405

- ## OBJECTIUS:

  Per aquesta pràctica hem d'implementar un menú sobre el programari realitzat a la pràctica 2, en aquest cas s'havia de modificar de tal manera de que funcionés dintre d'un model de menú que ens ha sigut facilitat *(plantilla-menu.c)* i, a més a més, s'havia d'implementar també una manera de guardar i carregar les dades amb les que treballa el programa.

- ## IMPLEMENTACIÓ:

  Per realitzar les tasques plantejades en aquesta pràctica he aplicat el codi de l'anterior entrega en les diferents opcions del menú. Com en l'anterior pràctica el codi no funcionava correctament, s'ha hagut de invertir temps en arreglar el codi de la anterior pràctica per tal de que anés correctament.

  Una vegada el codi de l'anterior pràctica funcionava a la perfecció i passava net el *Valgrind* he començat a implementar el principal objectiu per aquesta tercera entrega: la implementació del codi necessari per realitzar un guardat a disc del arbre que conté totes les dades dels aeroports i, al mateix temps, una funcionalitat que faci l'efecte invers, és a dir, una manera de carregar les dades, guardades en un fitxer que s'ha creat anteriorment, i que la resta de les funcionalitats que té aquest menú funcionin correctament amb aquestes dades carregades.

  Per tal d'assolir les funcions de guardar i carregar dades per i des de un fitxer he creat diferents funcions que realitzen aquestes tasques:

  1. ***int countTreeRecursive(node *n):*** Aquesta funció viatja per tot l'arbre comptant el número total de nodes que conté l'arbre de dades amb el que treballem durant tota la pràctica.

  2. ***void writeTree(rb_tree *tree, FILE *fp):*** Aquesta funció és cridada en l'opció dos del menú, on aquesta és l'opció de guardar l'arbre i totes les dades que conté, és a dir, els aeroports i les llistes de destinacions i vols. En executar-se, crida a la funció *writeTreeRecursive(node *n, FILE *fp),* passant com a node 'n' l'arrel de l'arbre.

  3. ***void writeTreeRecursive(node *n, FILE *fp):*** En aquesta funció entren per paràmetre un node del arbre i un punter de tipus FILE que apunta al fitxer que s'ha creat i obert on es guardaran totes les dades del arbre, primer de tot, en cas de que el node entrat per paràmetre tingui fills, farà una crida recursiva de si mateix amb el node fill juntament amb el punter FILE. Després cridarà a la funció *writeTreeData(node_data *n_data, FILE *fp)* on es realitzen les operacions de guardat de les dades.

4. ***void writeTreeData(node_data *n_data, FILE *fp):*** Amb aquesta funció s'aconsegueix guardar totes les dades del arbre dintre del fitxer en l'ordre en que s'especifica a la pràctica, aquest ordre és: Magic Number, Num. Nodes, Codi IATA, Num destinacions, llista de vols, etc.

   Per complementar aquesta funció i fer el guardat de la llista de vols per a cada node/aeroport he creat un altre funció nomenada *writeTreeList(list *l, FILE *fp).*

5. ***void writeTreeList(list *l, FILE *fp):*** Aquesta funció s'encarrega de recórrer la llista de cada node/aeroport i guardar al fitxer objectiu les dades de cada vol (destinació, retràs del vol).

Per la part de carregar les dades des de el fitxer, tot el codi està implementat directament en la opció del menú. Quan aquesta opció és seleccionada, primer de tot comprova que no hi hagi un arbre inicialitzat anteriorment, en cas positiu, l'esborra juntament amb totes les dades que aquest contenia.

Seguidament l'usuari entra per teclat el nom del fitxer i s'obre aquest fitxer amb un *fopen().* A continuació es llegeix el primer element del fitxer, que en aquest cas és el *magic number* que hem de tenir en consideració per comprovar que el fitxer que ha sigut carregat sigui del mateix tipus que el que utilitza el programa. En el cas de que no ho sigui, el programa mostra un missatge d'error per pantalla i atura l'execució d'aquest, sinó passa a llegir el segon element del fitxer. Ara aquest element és el nombre total de nodes que té arbre que s'ha emmagatzemat en aquest fitxer.

Després de fer les comprovacions anteriors, s'inicialitza l'arbre guardant espai dins de la memòria dinàmica i seguidament del *init_tree(tree).* Llavors carrega tots els nodes i les dades corresponents de cadascun (fent la reserva de espai de memòria dinàmica adequadament) i els insereix dintre de les llistes de cada node del arbre i, seguidament, els nodes dintre del propi arbre.

- ## COMPLICACIONS I OBSERVACIONS:

Respecte a complicacions, la que més problemes ha donat ha sigut el fet de que l'anterior pràctica no funcionava i per tant s'havia de completar correctament abans de poder realitzar la part corresponent a aquesta entrega.

Un altre de les complicacions que he tingut ha sigut en el moment de realitzat el guardat dels elements de la llista en que intentava a primeres guardar directament la llista de tots els vols de cada node però no funcionava i el *valgrind* no donava cap feedback al respecte.