2018-19

Sistemes Operatius II

PRÀCTICA 2

FRANCISCO DÍAZ RUIZ NIUB: 16828405

- **Comenteu els camps membres de l'estructura del node de l'arbre. Indiqueu quines funcions heu modificat (i com les heu modificat) per tal de permetre que l'arbre binari indexi per cadenes de caràcters. Feu el mateix pels camps membres de la llista indexada.**

En el cas de l'estructura dels nodes que té arbre, els camps que s'han modificat son els següents:

- En el fitxer *red-black-tree.h ,* en l'estructura (Struct *node_data_*) que defineix els *node_data*, s'ha afegit una llista enllaçada que contindrà els diferents vols que tindrà l'aeroport i, juntament amb aquesta llista, també s'ha afegit un comptador que indicarà el total de vols que tindrà cada aeroport. També s'ha modificat el tipus de *RBTREE_KEY_TYPE* de *int* a *char* .*
- Pel que fa al arxiu *red-black-tree.c,* s'han modificat algunes de les funcions que estaven definides. Primer de tot s'ha canviat la manera en que allibera el propi arbre, perquè abans només anava alliberant els nodes de manera iterativa, però ara que aquests nodes tenen a dintre també llistes dinàmiques, llavors s'han de alliberar prèviament abans de poder eliminar definitivament l'arbre.
  Després, en les funcions de comparació entre *keys (compare_key1_less_than_key2(), compare_key1_equal_to_key2())* s'han fet les modificacions adients per a que pugin comparar les claus ara que s'han modificat els seu tipus, sent abans *int* i ara *char*.* Ara aquestes dues funcions utilitzen la funció *strcmp(char*,char*)* de la llibreria *string.h,* en la que compara si es menor la primera clau que la segona es fa un retorn de si la comparació ha donat inferior a 0 i la que compara si són iguals retorna si la comparació ha donat igual a 0.
- A més, he afegit dos funcions noves que fan una cerca dintre del arbre per trobar l'aeroport amb més destinacions. Aquestes funcions són *search()* i *recursive_search().* La primera de les funcions retorna l'aeroport que té més destinacions després de haver cridat a la recursiva, on aquesta recorre tot l'arbre en busca del node amb el valor més alt de destinacions. Una vegada ja ha recorregut tot l'arbre, la funció *search()* retorna el resultat de la recerca.

Pel que fa a la llista, aquesta no ha tingut tants canvis, aquest són els següents:

- En el fitxer .c només s'ha modificat la manera en que es fa la comparació, ja que com ara estem treballant amb *char*,* llavors es pot utilitzar la funció *strcmp(char*,char*)* de la llibreria *string.h* i com ha de retornar si les dues claus són iguals, la funció ara retorna si la comparació ha donat 0.
- En el fitxer .h s'han modificat diverses coses, primer de tot s'ha canviat el tipus de les claus, passant de enters a *char*.* Després, a l'estructura de les *list_data* s'ha afegit un comptador de vols de tipus *int* i un comptador del retràs dels vols de tipus *double.*

- **Indiqueu algun dels resultats que obteniu amb el fitxer que se us proporciona amb la pràctica. Proveu, per exemple, amb els aeroports de DEN (el corresponent a Denver) o ATL (Atlanta).**

  He tingut diferents problemes a l'hora d'executar el codi. Primer de tot, encara que el codi compila perfectament i hauria de funcionar correctament, quan comença a omplir l'arbre, no se com s'ho fa però no deixa afegir nous nodes al arbre perquè diu que ja existeixen dintre de l'arbre, encara que aquest arbre està inicialment buit i no conté cap node al principi. Per tant no he pogut obtenir cap resultat. El codi que he fet es basa en que hauria de fer en cas de que no passés això.
  Tanmateix, com no he pogut obtenir els resultats esperats, tampoc he pogut provar a passar el Valgrind al codi, encara que crec que allibero tots els espais de memòria que s'han reservat a l'hora de realitzar la tasca.
  Si pot ser, m'agradaria tenir juntament amb la valoració de la pràctica una petita explicació dient on fallava si pot ser.