Sistemes Operatius II - Pràctica 1

Setembre del 2015

Aquesta primera pràctica dóna una visió global de les pràctiques a desenvolupar al llarg d'aquesta assignatura. S'indicaran també les tasques a desenvolupar en la primera pràctica. La data d'entrega de la primera pràctica és dimarts 5 d'octubre pel grup F i dimecres 7 d'octubre pel grup A i C.

Índex

1 Introducció **2**

2 La pràctica **2**

2.1 Lectura de les paraules del fitxer 3

2.2 Inserció de les paraules en un arbre binari 3

3 Implementació i planificació **4**

3.1 Lectura de dades d'un fitxer 4

3.2 Inserció de paraules en un arbre binari 5

4 Entrega **5**

5 Alguns detalls sobre la pràctica 2 **6**

1 Introducció

A l'assignatura de Sistemes Operatius II es realitzarà un únic projecte pràctic al llarg del curs. L'objectiu general del projecte pràctic és desenvolupar una aplicació (sense interfície gràfica) que permeti extreure i indexar les paraules contingudes en múltiples fitxers de text pla. Els fitxers de text són proporcionats pel professor tot i que també es poden baixar lliurement d'Internet. Aquests fitxers de text són llibres electrònics gratuïts en anglès i provenen del web Gutenberg (http://www.gutenberg.org/). L'aplicació haurà de llegir cadascun d'aquests fitxers de text, extreure i indexar totes les paraules que contenen aquests (pràctica 1 i 2). Un cop s'ha obtingut aquesta informació s'haurà d'analitzar i visualitzar determinades parts de la informació extreta (pràctica 3). Finalment, per fer l'aplicació més ràpida s'aprofitaran els múltiples processadors que ens ofereixen les màquines del laboratori per tal de paral·lelitzar determinades tasques (pràctica 4).

Les pràctiques es realitzen en parella i podeu discutir amb els vostres companys la solució que implementeu. En tot cas, s'espera que cada parella implementi el seu propi codi.

Les pràctiques es realitzaran en llenguatge C i estaran centrades en els següents aspectes del llenguatge:

- Pràctica 1: punters i estructura de dades (entrega dilluns 5 o dimecres 7 d'octubre segons grup de pràctiques)
- Pràctica 2: manipulació de cadenes de caràcters i lectura de dades de fitxers (entrega dilluns 26 o dimecres 28 d'octubre segons grup de pràctiques).
- Pràctica 3: comunicació interprocés mitjançant canonades i escriptura de dades a disc (entrega dilluns 23 o dimecres 25 de novembre segons grup de pràctiques).
- Pràctica 4: programació multifil (entrega dimecres 23 de desembre per a tots els grups de pràctiques)

Les pràctiques estan encavalcades entre sí. És a dir, per a realitzar una pràctica és necessari que la pràctica anterior funcioni correctament. Assegureu-vos doncs que les pràctiques estan dissenyades de forma que puguin ser ampliades de forma fàcil (més endavant es donaran algunes idees per aconseguir-ho).

2 La pràctica

L'objectiu de la primera pràctica és manipular punters i estructures de dades. També hi ha una part petita de lectura de fitxers de disc. Específicament, l'objectiu d'aquesta primera pràctica és:

1. Llegir d'un fitxer de text pla un conjunt de paraules en anglès. A cada línia del fitxer hi haurà una única paraula. Els detalls d'aquest punt s'expliquen a la secció 2.1.
2. Inserció de les paraules en un arbre binari. Veure secció 2.2.

Els dos punts anteriors es descriuen a continuació. A la secció 5 es descriu breument què és el que haureu de fer a la pràctica 2. D'aquesta forma podeu estructurar el programa de la pràctica 1 en funció de la feina que haureu de fer més endavant.

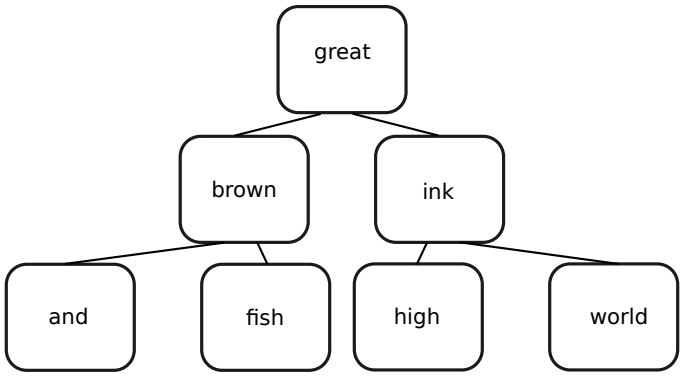

Figura 1: Arbre binari que emmagatzema paraules.

2.1 Lectura de les paraules del fitxer

A les aules hauríeu de tenir instal·lada una aplicació per terminal anomenada `look` que permet cercar paraules dintre d'un fitxer de diccionari. Per exemple, executeu

```
$ look see
```

Aquesta instrucció cerca, dintre del fitxer de dicccionari (per defecte a `/usr/share/dict/words`), totes les paraules que comencen per `see` i les mostra per pantalla.

Disposeu, juntament amb l'enunciat d'aquesta pràctica, del fitxer de diccionari `words`. Aquest fitxer es troba en el directori `src/llegir-diccionari`. Aquest fitxer és un fitxer que conté paraules de diccionari en anglès. Les paraules estan separades entre sí per línies (editeu-lo amb un editor de text pla!).

El fitxer de què disposeu amb aquesta pràctica ha estat netejat per eliminar totes les paraules que contenen accents o dièresis. Les paraules amb accents o dièresis s'emmagatzemen a un fitxer de text fent dos bytes en comptes d'un byte com és el cas de les lletres 'a' a 'z' i 'A' a 'Z'. D'aquesta forma, en eliminar les paraules que contenen accents o dièresis, es facilita el tractament de fitxers de text en llenguatges com el C.

La lectura de dades d'un fitxer és un tema delicat ja que hi ha múltiples funcions que permeten fer-ho. Hi entrarem amb més detall a la pràctica 2. En aquesta pràctica només es demana llegir la paraula que hi ha emmagatzemada a cada línia. La forma més senzilla de fer-ho és fer servir la funció *fscanf*, vegeu la secció 3 per veure'n una implementació.

2.2 Inserció de les paraules en un arbre binari

L'objectiu en aquesta pràctica és inserir les paraules llegides del fitxer `words` en un arbre binari. Observeu la figura 1. És un exemple d'arbre binari que conté paraules (cadenes de caràcters). Per a cada node s'emmagatzemen a l'esquerra les paraules que lexicogràficament són anteriors a la paraula del node, mentre que a la dreta es troben les paraules lexicogràficament posteriors a la paraula del node.

3 Implementació i planificació

Per tal d'assolir amb èxit aquesta pràctica es recomana revisar abans de tot la Fitxa 1 (recordatori bàsic del llenguatge C i manipulació de cadenes de caràcters). És particularment important la secció "Manipulació de cadenes caràcters". A l'hora de programar és important seguir una estructura modular atès que la resta de pràctiques d'aquesta assignatura es basaran en aquesta primera pràctica. Es proposa a continuació una forma de procedir per la implementació d'aquesta pràctica.

3.1 Lectura de dades d'un fitxer

A la secció 2.1 s'ha proposat utilitzar la funció *fscanf* per llegir el fitxer. Per a la funcionalitat que es necessita en aquesta pràctica és suficient. Cal tenir en compte, però, que aquesta funció no és gaire segura ja que no permet controlar bé el nombre de bytes que es llegeixen en llegir una cadena de caràcters (ho veurem a la pràctica 2). Per comprovar això es proposen realitzar els següents experiments:

1. Al directori `src/llegir-diccionari` trobareu el diccionari de paraules a llegir així com un codi C que llegeix aquestes paraules. Executeu aquest codi i comproveu que funciona correctament. Per compilar podeu utilitzar aquesta instrucció

   ```
   $ gcc fitxer-fscanf.c -o fitxer-fscanf
   ```

2. Editeu el codi i modifiqueu el valor de `MAXCHAR`. Per exemple, utilitzeu un valor de 5. Abans d'executar el codi, penseu-hi... què creieu que passarà? Petarà el programa en llegir la primera paraula de més de 5 caràcters? Proveu-ho compilant i executant el codi. Observeu si el programa peta en algun moment i comproveu si abans de petar ha llegit alguna paraula de més de 5 caràcters. Sou capaços de trobar una explicació a aquest comportament?

3. A continuació executeu l'aplicació amb el `valgrind` (vegeu transparències) per veure si l'aplicació és capaç de detectar quan es produeix per primer cop el problema. Per això cal compilar el codi amb opció de "debugger"

   $ `gcc -g fitxer-fscanf.c -o fitxer-fscanf`

   Executeu a continuació l'aplicació amb la següent comanda

   ```
   $ valgrind ./fitxer-fscanf words 2>&1 | less
   ```

   Aquesta comanda fa que s'imprimeixin per pantalla (i de forma interactiva) els errors que detecta `valgrind`. Comenceu pel primer error. Què us indica? A quina línia es produeix l'error? El missatge d'error imprès per `valgrind` també us indica en quina línia s'ha fet el *malloc* corresponent. On està indicat això?

4. La darrera prova es basa en modificar el codi font i declarar la variable cadena com estàtica. És a dir, modifiqueu el codi perquè es declari com

   ```
   char paraula[MAXCHAR];
   ```

   Compileu el codi i executeu el codi amb el `valgrind`. Què passa ara? Observeu en quin moment detecta `valgrind` el problema. Idealment, `valgrind` hauria de trobar el problema en el mateix moment que en el punt 3. És així? En podeu treure alguna conclusió d'aquest experiment? Per trobar ràpidament els problemes al codi, què és millor: fer servir vectors dinàmics o vectors estàtics?

3.2 Inserció de paraules en un arbre binari

Cal inserir cada paraula llegida del fitxer `words` en un arbre binari. Al directori `src/arbre-binari` disposeu del codi que fareu servir en aquestes pràctiques. En concret, els fitxers `red-black-tree.c` i `red-black-tree.h` contenen la implementació d'un arbre binari balancejat. El fitxer `main.c` conté un exemple d'ús pel cas en què es vulguin inserir sencers a l'arbre. Es proposa doncs

1. Executeu el codi d'exemple. Per compilar-lo cal executar `make` dins del directori en què es troba el fitxer `Makefile`.

2. Editeu i analitzeu el funcionament de l'exemple, en concret de la funció `main`. Observeu com s'insereixen nous nodes a l'arbre i com es tracta el cas en què un node ja existeix a l'arbre.

3. Modifiqueu el codi per adaptar-lo a les necessitats d'aquesta pràctica: en comptes d'inserir sencers a l'arbre caldrà inserir paraules (cadenes de caràcters). Per això caldrà editar els fitxers `red-black-tree.h` i `red-black-tree.c`, modificar les estructures corresponents, així com algunes de les funcions. Us serà particularment útil la funció *strcmp*, que permet comparar dues cadenes de caràcters. Llegiu atentament al manual d'aquesta funció (la trobareu a Internet, per exemple) per saber com funciona.

4. Un cop modificat el codi integreu-lo amb la lectura de les paraules de la subsecció anterior. Abans d'inserir una paraula de diccionari a l'arbre heu de passar tots els seus caràcters a minúscula. Les funcions *isupper* i *islower* permeten saber si una lletra és majúscula o minúscula i les funcions *toupper* i *tolower* permeten convertir una lletra a majúscula o minúscula respectivament.

5. L'aplicació ha d'imprimir, en finalitzar l'execució, totes les paraules emmagatzemades a l'arbre. Comproveu que el nombre de nodes de l'arbre coincideix amb el nombre de línies del fitxer. Assegureu-vos a més que tot funciona correctament fent servir el `valgrind`.

4 Entrega

El fitxer que entregueu s'ha d'anomenar `P1_NomCognom1NomCognom2.tar.gz` (o `.zip`, o `.rar`, etc), on `NomCognom1` és el nom i cognom del primer component de la parella i `NomCognom2` és el cognom del segon component de la parella de pràctiques. El fitxer pot estar comprimit amb qualsevol dels formats usuals (`tar.gz`, `zip`, `rar`, etc). Dintre d'aquest fitxer hi haurà d'haver dues carpetes: `src`, que contindrà el codi font, i `doc`, que contindrà la documentació addicional en PDF. Aquí hi ha els detalls per cada directori:

- El directori `doc` ha de contenir un document (tres o quatre pàgines, en format PDF, sense incloure la portada) explicant:

  - A la secció 3 es realitzen una sèrie de preguntes. Utilitzeu aquestes preguntes com a fil per tal de realitzar el vostre document. Intenteu no respondre a les qüestions directament, com si fos un examen.
  - Comenteu també breument el funcionament de la vostra aplicació. Feu servir un diagrama de blocs, per exemple. No entreu en detalls ja que aquests es poden veure en llegir el codi.

- La carpeta `src` contindrà el codi font comentat (com a mínim les funcions). S'hi han d'incloure tots els fitxers necessaris per compilar i generar l'executable. El codi ha de compilar sota Linux amb la instrucció `make`. Editeu el fitxer `Makefile` en cas que necessiteu afegir fitxers C que s'hagin de compilar.

La data límit d'entrega d'aquesta pràctica és dimarts 5 pel grup F i dimecres 7 d'octubre pel grup A/C. El codi té un pes d'un **50%** (codi amb funcions comentades, codi modular i net, ús correcte del llenguatge, bon estil de programació, el programa funciona correctament, tota la memòria és alliberada, sense accessos invàlids a memòria, etc.). El document i les proves tenen un pes del **50%** restant.

5 Alguns detalls sobre la pràctica 2

A la segona pràctica se us proporcionaran uns 600 fitxers de text pla. Cadascun d'aquests fitxers és el text d'un llibre en anglès. El treball a realitzar serà

1. Extreure cadascuna de les paraules que hi ha als fitxers. A cada línia d'aquests fitxers de text hi pot haver múltiples paraules (i no com en aquesta pràctica, en què hi ha una paraula per cada línia). Per realitzar aquesta tasca se us proporcionarà un codi que permet extreure les paraules del fitxer.

2. Per a cada fitxer es farà servir una taula en què s'inseriran les paraules extretes. En aquesta taula es comptabilitzarà el nombre de vegades que apareix cadascuna de les paraules al fitxer.

3. Un cop finalitzat l'extracció de paraules d'un fitxer es procedirà a la inserció de la informació a l'arbre creat en aquesta primera pràctica. Només es traspassarà la informació d'aquelles paraules que existeixen a l'arbre. Totes les paraules que siguin a la taula però que no existeixin a l'arbre s'ignoraran.