Universitat de Barcelona

Sistemes Operatius II

Pràctica 1: Extracció de paraules d'un fitxer

Igor Dzinka, Vicent Roig

03/10/2014

Què es vol fer a la pràctica.

L'objectiu de la primera pràctica és aprendre a manipular cadenes de caràcters i a llegir dades de fitxers. Específicament, l'objectiu d'aquesta primera pràctica és:

1. Aprendre les diverses tècniques amb què es poden llegir dades d'un fitxer. Es pretén que analitzeu les avantatges i desavantatges de cadascuna de les tècniques per tal d'escollir la que millor us convingui per a la implementació de la pràctica.

2. . Implementar un algorisme que permeti extreure les paraules que hi ha en un únic fitxer de text pla (el problema d'analitzar múltiples fitxers de text es deixarà per a la segona pràctica)

Qüestions plantejades.

1. Lectura byte a byte (mitjançant la funció fgetc).

A l'exemple de la figura 2 se us mostra un exemple d'ús. Compileu i executeu aquest codi. Teniu un exemple de fitxer de text pla juntament amb el codi font. Observeu què és el que surt per pantalla. En executar apareix, de tant en tant, **el caràcter amb codi ASCII 10. A quin caràcter correspon?**

```
==5515== Command: ./exemple-fgetc
==5515==
Caracter: H (byte 72)
Caracter: o (byte 111)
Caracter: l (byte 108)
Caracter: a (byte 97)
Caracter:   (byte 32)
Caracter: h (byte 104)
Caracter: O (byte 79)
Caracter: l (byte 108)
Caracter: A (byte 65)
Caracter:   (byte 32)
Caracter: H (byte 72)
Caracter: O (byte 79)
Caracter: L (byte 76)
Caracter: A (byte 65)
Caracter: ! (byte 33)
Caracter:
 (byte 10)
Caracter: B (byte 66)
Caracter: o (byte 111)
Caracter: n (byte 110)
Caracter: . (byte 46)
Caracter: d (byte 100)
Caracter: i (byte 105)
Caracter: a (byte 97)
Caracter: . (byte 46)
```

El caràcter amb codi ASCII 10 correspon al salt de línia '\n'

Observeu que el procediment per llegir el fitxer és:

a) llegir el caràcter. b) comprovar si s'ha arribat a final de fitxer i c) imprimir el caràcter per pantalla. **Per què cal seguir aquest ordre i no podem fer la comprovació b) abans de fer a) ?**

La funció fgetc retorna el caràcter apuntat per un punter de posició dins del fitxer. Un cop llegit el caràcter el punter avança una posició al caràcter següent. En el cas de que el punter es trobi a End Of File quan s'ha fet la crida a fgetc, la funció retorna EOF i activa l'indicador de End Of File (feof). Per aquest motiu ens interessa primerament fer la lectura del caràcter, cosa que ens permetrà determinar si hem arribat al End of File, I, posteriorment, fer la comprovació, ja que tindrem el valor de feof actualitzat.

Per a la aplicació prevista (extreure les paraules**), penseu que es tracta d'una bona forma de llegir el fitxer ? És una forma eficient d'abordar el problema ?**

De cara a la realització de la pràctica, es pot aconseguir l'objectiu fent ús de la funció fgetc. Però el problema que te aquest mètode es que realitzarem un gran nombre d'accessos al fitxer, i cada consulta al fitxer esdevé una crida al sistema. Això fa que es ralentitzi el funcionament de la CPU. Com que nosaltres no fem servir fitxers de text gaire grans això passa desapercebut però amb fitxers grans això es farà evident.

2. Lectura del fitxer línia a línia (mitjançant la funció fgets).

A l'exemple de la figura 3 se us mostra l'exemple que llegeix línia a línia fent servir fgets. Observeu de nou que el procediment per llegir el fitxer és: a) llegir la línia, b) comprovar si s'ha arribat a final de fitxer i c) imprimir la línia per pantalla. En executar el codi, observeu que no apareixen totes les línies del fitxer. **Sou capaços de modificar el codi font perquè apareguin totes les línies però sense modificar el fitxer ?**

```
==6544== Command: ./exemple-fgets
==6544==
Linia: Hola hOlA HOLA!

Linia: Bon.dia.prova, prova*# grau enginyeria **

Linia: paraula1 paraula2 paraula3 paraula 4 414 FN212 FN21N

Linia: Good Good-good Good's Göood good@ub.edu

Linia: link http://www.ub.edu

Linia: fine!
```

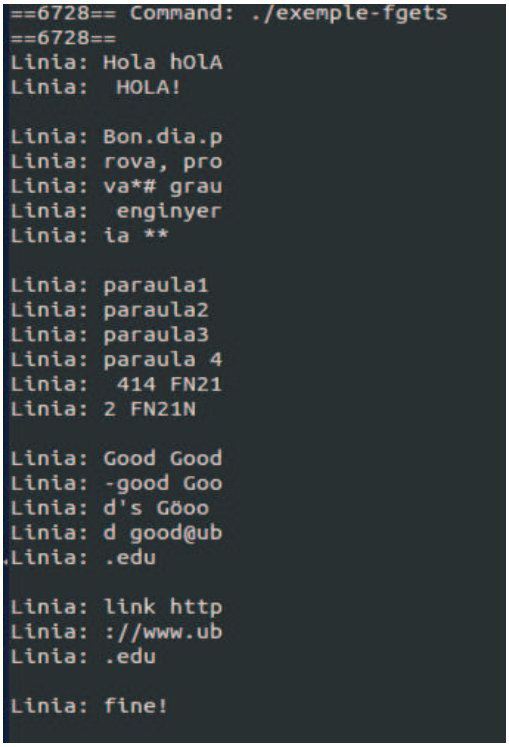
Si, la modificació seria la següent:

```
while( fgets(str,100,fp)!=NULL ){

       printf("Linia: %s\n", str);

}
```

Cal tenir en compte que fgets retorna NULL si arribem al final del fitxer, així que podem iterar fins que la funció retorni aquest valor per determinar quan hem acabat de llegir-ho tot.

Tingueu en compte que, tal i com es comenta a la fitxa 2, la funció fgets llegeix del fitxer fins arribar al caràcter final de línia ('\n') tenint en compte que a tot estirar ha de llegir un determinat nombre de bytes especificat a l'argument de fgets, que en aquest exemple és 100.

Modifiqueu el codi perquè es llegeixin un màxim de 10 bytes. El que surt per pantalla és el que esperàveu ?



El resultat d'executar fgets amb el màxim de lectura de 10 caràcters es el que es mostra a la imatge següent imatge:

Com la funció ara es limita a agafar conjunts de 10 caràcters, no te capacitat per mostrar tota una línia d'un cop. Per tant el que tenim és que es van mostrant per pantalla conjunts de 10 caràcters fins arribar al final de línia.

Què penseu d'utilitzar fgets per llegir el fitxer ?

És una forma raonable d'arribar a un compromís entre la memòria a ocupar i les dades a processar.

En la nostra opinió; l'ús de la funció fgets permet solucionar alguns dels inconvenients de les funcions fgetc i fread, equilibrant el cost d'accessos i memòria. Per un costat, realitza moltes menys crides al sistema en comparació a la funció fgetc, això permet que es realitzi mes eficientment la lectura, sobretot en un fitxer amb moltes línies. Per altra banda, no ens obliga a carregar tot el contingut (el qual pot ser molt gran i probablement no processarem tot alhora) a memòria, com és el cas d'utilitzar la funció fread. Realment ens permet llegir fragments seqüencials del fitxer i tractar-los per parts.

En el cas de llegir 100 bytes cada cop, tingueu en compte que fgets inclou el caràcter '\n' a la cadena de caràcters (ja que cada línia del fitxer d'exemple té una longitud inferior a 100 caràcters**). Quines instruccions C faríeu servir per tal "d'eliminar" el caràcter '\n' del final de la cadena ?**

Una de les possibles solucions a aquest problema seria fer l'us de la següent funció:

```
strtok(string, "\n");
```

Una forma menys elegant:

```
char *pos;

if ((pos=strchr(Name, '\n')) != NULL)

        *pos = '\0';
```

3. Lectura mitjançant la funció fscanf.

Una forma típica de llegir un fitxer formatat és mitjançant la funció fscanf. A la figura 4 se us mostra la forma de procedir. **Sembla que fscanf està separant "automàticament" el fitxer d'entrada en paraules. Com ho està fent exactament?**

La funció fscanf, per defecte llegeix dades d'un flux de dades (stream) donat un format determinat. Si llegim una cadena (%s), la lectura ignora tots els elements de tipus Whitespace (espais en blanc, salts de línia, tabulacions). Cal fixar-se però, que el seu comportament ens permet ajustar la lectura al format que nosaltres determinem.

Precisament, la gràcia de la funció fscanf és que, a diferència de les altres que hem vist, ens permet adequar la lectura del fitxer a un tipus determinat de variable. De forma que si coneixem les dades emmagatzemades al fitxer, podrem carregar-les a una variable del tipus adient (%d per integers, %o per nombre en format octal, %u per unsigned integer, %s per cadenes de caràcters, %c per lectura d'un caràcter, etc.)

L'ús de fscanf us facilitarà la implementació de l'algorisme d'extracció de paraules respecte les altres solucions proposades ? Raoneu la vostra resposta.

```
==6961== Command: ./exemple-fscanf
==6961==
Dades llegides: Hola
Dades llegides: hOlA
Dades llegides: HOLA!
Dades llegides: Bon.dia.prova,
Dades llegides: prova*#
Dades llegides: grau
Dades llegides: enginyeria
Dades llegides: **
Dades llegides: paraula1
Dades llegides: paraula2
Dades llegides: paraula3
Dades llegides: paraula
Dades llegides: 4
Dades llegides: 414
Dades llegides: FN212
Dades llegides: FN21N
Dades llegides: Good
Dades llegides: Good-good
Dades llegides: Good's
Dades llegides: Göood
Dades llegides: good@ub.edu
Dades llegides: link
Dades llegides: http://www.ub.edu
Dades llegides: fine!
```

A primera vista, la funció fscanf sembla de gran utilitat de cara a la implementació de l'algorisme sol·licitat a la practica, donat que ens permet extreure les paraules típicament separades per espais del fitxer. Només tindríem que tractar cadascuna d'aquestes paraules i passar-les per un procés de validació per descartar aquelles que no compleixin els requisits.

*Però si ho analitzem més en profunditat, com la funció bàsicament realitza la separació de paraules al localitzar-ne de tipus whitespace (espais en blanc, tabulacions, salts de línia). Seguim obtenint tot un seguit de paraules que no són interessants, com podem veure a la imatge en el cas particular de "**" entre d'altres. Al final tindrem que processar cadascuna d'aquestes paraules per comprovar si efectivament és una paraula vàlida, brossa, o cal tractar alguns dels caràcters inclosos no desitjats.*

Per tant, podem concloure que encara que sembla una opció més adient que fgetc, al contribuir d'alguna manera a agilitzar el procés de detecció de les paraule i realitzant menys crides a sistema. Seguim sense obtenir una milloria significativa respecte a l'opció de carregar línies de text limitades a N caràcters (fgets) per posteriorment, poder-les tractar caràcter a caràcter i determinar si és o no una paraula.

A l'exemple s'ha suposat que la longitud màxima de la paraula és de 20 caràcters. **Què passarà si al fitxer hi ha una paraula amb una longitud superior a 20 caràcters ?**

Si es donés el cas de que el fitxer contingués una paraula de mida superior a 20 caràcters, la funció intentarà carregar a memòria els caràcters restants fora de l'espai que té assignat (bytes consecutius), cosa que pot produir un error de segmentació.

Lectura del fitxer amb la funció fread. La solució és la que es proposa a la figura 5. **Què en penseu d'aquesta solució ?**

```
==2663==
Hola hOlA HOLA!
Bon.dia.prova, prova*# grau enginyeria **
paraula1 paraula2 paraula3 paraula 4 414 FN212 FN21N
Good Good-good Good's Göood good@ub.edu
link http://www.ub.edu
fine!
```

Fent servir la lectura mitjançant fread, permet carregar tot el contingut del fitxer amb una sola crida. Com que no tractem fitxers gaire grans a aquesta pràctica, aquesta es una solució factible per portar les dades a memòria I un cop fet això fer-ne el tractament. Però cal tenir en compte que els fitxer poden ser molt grans i no és una bona pràctica carregar d'un sol cop un conjunt molt gran de dades a memòria, quan podem anar fraccionant les càrregues a mesura que les tractem.

En el cas hipotètic que el fitxer a llegir sigui molt gran és útil aquesta solució ?

En el cas de que intentéssim portar d'un cop el contingut d'un fitxer massa gran, podríem tenir alguns problemes. Per fer la lectura hem de fer una crida al sistema. aquest atura la seva activitat durant la lectura queda a l'espera de finalitzar l'operació. Si es tracta d'una lectura molt llarga el sistema es quedaria molt de temps parat, cosa que pot produir errors al funcionament del equip.

Un altre possible problema es que no hi hagi espai suficient a la memòria per guardar tot el contingut llegit.

Implementació de l'algoritme

En aquesta primera pràctica l'objectiu va ser aconseguir un algorisme que extregui les paraules d'un fitxer de text I imprimir-les per pantalla, tot dient si son paraules vàlides o no.

Tal i com em anat comentant durant les proves i consegüents respostes, finalment em decidit optar per anar llegint del fitxer mitjançant fgets. ja que la considerem la més óptima de totes per realitzar la tasca. No fa tantes crides al sistema com la funció fgetc ni causa problemes amb la memoria que pot causar la funció fread. No hem optat per la funció fscanf ja que no realitza la separació de paraules de manera correcta I no hem vist oportú el seu ús.

Hem establert un límit per línia de 100 caràcters i un altre de 50 caràcters per emmagatzemar una paraula segons estimacions. Aquest últim array l'utilitzarem per anar construint les paraules de forma seqüencial.

*Aquests arrays romandran a la zona de memòria dinàmica el temps necessari per fer-ne ús alliberant la memòria quan sigui pertinent. A més em modulat l'extracció de paraules per línia a una funció **findWords.***

Un cop hem extret la la línia de text la recorrem caràcter a caràcter per poder extreure les paraules. Tots els caràcters que siguin espais tabulacions Si una paraula conté algun tipus de caràcter alfanumèric o d'un altre tipus (excepte l'apòstrof que per les especificacions de la pràctica es considera un caràcter vàlid) aquesta paraula es considera no vàlida.

Resultat d'execució per fitxer.txt:

```
Paraula valida: hola
Paraula valida: hola
Paraula valida: hola
Paraula valida: bon
Paraula valida: dia
Paraula valida: prova
Paraula valida: prova
Paraula valida: grau
Paraula valida: enginyeria
Paraula invalida: paraula1
Paraula invalida: paraula2
Paraula invalida: paraula3
Paraula valida: paraula
Paraula invalida: 4
Paraula invalida: 414
Paraula invalida: fn212
Paraula invalida: fn21n
Paraula valida: good
Paraula valida: good
Paraula valida: good
Paraula valida: good's
Paraula valida: göood
Paraula valida: good
Paraula valida: ub
Paraula valida: edu
Paraula valida: link
Paraula valida: http
Paraula valida: www
Paraula valida: ub
Paraula valida: edu
Paraula valida: fine
```