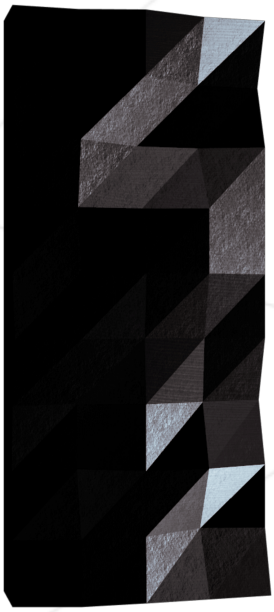

Proiect modul Algoritmica I

get_next_line

Staff staff@42.fr

Sumar: Acest proiect are ca obiectiv sa scrieti o functie ce returneaza o linie ce se termina cu un retur de linie de la un descriptor de fisier.

Cuprins

Capitolul I

Preambul

Suave, mari magno turbantibus aequora ventis,
e terra magnum alterius spectare laborem ;
non quia vexari quemquamst jucunda voluptas,
sed quibus ipse malis careas quia cernere suave est.
Suave etiam belli certamina magna tueri
per campos instructa, tua sine parte pericli.
Sed nihil dulcius est bene quam munita tenere
edita doctrina sapientum templa serena
despicere unde queas alios passimque videre
errare, atque viam palantis quaerere vitae,
certare ingenio, contendere nobilitate,
noctes atque dies niti praestante labore
ad summas emergere opes rerumque potiri.
Ô miseras hominum mentes, ô pectora caeca!
Qualibus in tenebris vitae quantisque periclis
degitur hoc aevi quodcumque est ! Nonne videre
nihil aliud sibi naturam latrare, nisi ut qui
corpore sejunctus dolor absit, mensque fruatur
jucundo sensu cura semota metuque?

Un pic de cultura, dragilor !

Capitolul II

Subiect

II.1 Parte obligatorie

- Scrieti o functie ce returneaza o linie luata dintr-un descriptor de fisier.
- Numim “liie” o succesiune de caractere terminate printr-un ’\n’ (cod ascii 0x0a) sau printr-un End Of File (EOF).

https://latedev.wordpress.com/2012/12/04/all-about-eof/: acest articol este relevant pentru cei ce stiu sa citeasca.

- Functia voastra va trebui sa aiba prototipul urmator:

```
int     get_next_line(int const fd, char ** line);
```

- Primul parametru este descriptorul de fisier din care se citeste.
- Al doilea parametru este adresa unui pointer spre caracterul ce serveste la stocarea linia luata din descriptorul de fisier.
- Valoarea returnata poate fi 1, 0 sau -1 daca o linie a fost citita, citirea s-a terminat, sau s-a produs o eroare.
- Functia get_next_line trebuie sa returneze rezultatul fara ’\n’.
- Un apel in bucla a functiei get_next_line va permite deci, citirea textului disponibil in descriptorul de fisier, linie cu linie, pana la sfarsitul textului.
- Asigurati-va ca functia se comporta bine cand citeste dintr-un fisier, de la intrarea standard, dintr-o redirectare etc.

II.2 Livrare

- Trebuie sa livrati in radacina directorului vostru de lucru/livrare, un fisier `auteur` continand login-ul vostru urmat de '\n':

```
$>cat -e auteur
xlogin$
$>
```

- Trebuie sa livrati doua fisiere: `get_next_line.c` si `get_next_line.h`
- Nu trebuie sa aveti nici o functie `main` in directorul vostru de lucru/livrare.
- Nu livrati fisier `Makefile`.
- Continutul prezent in directorul vostru de lucru/livrare va fi evaluat la sustinere.

II.3 Considerente tehnice

- Fisierul vostru `get_next_line.h` trebuie cel putin sa contina prototipul functiei `get_next_line` si un macro ce permite dimensiunea bufferului de citire la fiecare apel `read`. Aceasta valoare va fi modificata la sustinere pentru evaluarea calitatii livrarii voastre. De exemplu:

```
#define BUFF_SIZE 32
```

Codul vostru functioneaza intotdeauna cand BUFF_SIZE ia valoarea 9999? Si cand BUFF_SIZE ia valoarea 1 ? Si 10000000? Stiti de ce?

- Se considera ca functia `get_next_line` are un comportement nedeterminat intre doua apeluri, cand un acelasi descriptor de fisier desemneaza doua fisiere diferite in momentul in care citirea primului fisier nu e terminata.

- De asemenea, consideram ca un apel la functia `lseek(2)` nu va avea loc niciodata intre doua apeluri ale `get_next_line` pe acelasi descriptor de fisier.

- In fine, consideram ca `get_next_line` are un comportement nedeterminat in cazul in care se citeste dintr-un fisier binar. Astfel, daca doriti, puteti adauga aceasta functionalitate.

- Este interzis sa folositi variabile globale pentru salvarea caracterelor ce au fost citite, dar care nu au fost returnate de functie. Puteti folosi variabile statice.

Daca stiti ce este o variabila statica este un bun inceput:
https://en.wikipedia.org/wiki/Static_variable

- Daca sunteti isteti si folositi biblioteca `libft` pentru functia `get_next_line`, trebuie sa copiati sursele si `Makefile`-ul asociat intr-un director numit `libft` ce va trebui sa fie in radacina directorului vostru de lucru/livrare. La sustinere, cel care va corecteaza va compila livrarea in felul urmator (fisierul `main.c` va fi a celui care va corecteaza):

```
$>make -C libft/ fclean
$>make -C libft/
$>gcc -Wall -Wextra -Werror -I libft/includes/ -c get_next_line.c
$>gcc -Wall -Wextra -Werror -I libft/includes/ -c main.c
$>gcc -o test_gnl get_next_line.o main.o -L libft/ -lft
```

- Daca dimpotriva, nu utilizati biblioteca `libft` la functia `get_next_line`, cel care va corecteaza va compila livrarea in felul urmator, la sustinere (fisierul `main.c` va fi a celui ce va corecteaza):

```
$>gcc -Wall -Wextra -Werror -c get_next_line.c
$>gcc -Wall -Wextra -Werror -c main.c
$>gcc -o test_gnl get_next_line.o main.o
```

II.4 Functii autorizate

- `read`
- `malloc`
- `free`

II.5 Bonus

Proiectul `get_next_line` este simplu si lasa putine optiuni pentru adaugarea de bonusuri, dar sunt convins ca aveti multa imaginatie. Daca veti reusi sa realizati perfect partea obligatorie, aceasta sectiune va propune cateva piste pentru a merge mai dearte. Atentie, niciun bonus nu va fi validat daca la partea obligatorie nu obtineti cel putin 18 din cele 20 puncte.

- Sa realizati functia `get_next_line` cu o singura variabila statica.
- Puteti folosi mai multi descriptori de fisier pentru `get_next_line`. De exemplu, daca descriptorii de fisier 3, 4 si 5 sunt accesibili la citire, atunci se poate apela `get_next_line` odata pentru 3, odata pentru 4 si din nou pentru 3, apoi odata pentru 5 etc, fara a pierde firul citirii pe fiecare din descriptori.

Capitolul III

Instructiuni

- Acest proiect nu va fi corectat doar de fiinte umane.
- Proiectul trebuie sa fie scris conform standardului de cod (Norme). Norminette va fi utilizata pentru verificarea Normei, care se aplica in ansamblu.
- Va trebui sa tratati erorile intro maniera adecvata. In niciun caz programul nu trebuie sa se opreasca intr-un mod neasteptat (Eroare de segmentare, eroare de magistrala etc).
- Toata memoria alocata va trebui eliberata in mod adecvat cand este necesar.
- Trebuie sa livrati in radacina directorului vostru de lucru/livrare, un fisier `auteur` continand login-ul vostru urmat de '\n':

```
$>cat -e auteur
xlogin$
$>
```

- Daca decideti sa livrati acest proiect folosind biblioteca `libft`, va este formal interzis sa adaugati functii specifice la livrarea functiei `get_next_line` ca sa evitati constrangerile impuse de standardul de cod (Norme). Aceasta va fi considerata o tentativa de trisare la sustinere. Functia voastra `get_next_line` trebuie sa fie compusa din cel mult 5 functii de maxim 25 de linii. Respectarea acestei constrangeri va fi in mod minutios verificat la sustinere. Este inutil sa incercati sa convingeti membri staff-ului daca o anumita functie ar fi acceptabila si pe care vreti sa o adaugati bibliotecii voastre. Intrebati-va mai bine daca functia voastra incalca sau nu aceste constrangeri dupa constiinta voastra. Daca veti respecta aceasta regula veti fi incurajati sa va extindeti biblioteca cu functii generice pe a caror utilitate ati descoperit-o pe parcursul acestui proiect.

Capitolul IV

Notare

Notarea functiei `get_next_line` se face in doi timpi:

- In primul rand se va testa partea obligatorie. Aceasta va fi notata cu maxim 20 puncte.
- Apoi, vor fi evaluate bonusurile. Ele vor fi notate cu maxim 5 puncte.
- Bonusul va fi evaluat doar daca la partea obligatorie ati obtinut cel putin 18 din 20 puncte.

Succes tuturor!