Tema 3 - Simple Linux File System

Responsabili:

- Luca Istrate [mailto:luca.istrate@gmail.com]
- Dănuț Matei [mailto:matei.danut.dm@gmail.com]
- Horia Ignat [mailto:ignat.horia.andrei@gmail.com]

- Data publicării: **08.05.2022 18:00**
- Deadline **HARD**: **28.05.2022 23:55:00**

Actualizări

- Adaugat tema vmchecker **25.05.2022 17:25:00**
- Corectat teste greșite **17.05.2022 20:56:00**
- Adăugat **checker** *cu teste* **16.05.2022 22:25:00**
- Adăugat **checker 15.05.2022 19:22:00**
- Publicat temă **8.05.2022 18:00:00**

Obiective

- Aprofundarea cunoștințelor în utilizarea limbajului C.
- Implementarea și utilizarea structurii de date **arbore [https://en.wikipedia.org/wiki/Tree_(data_structure)]** .

Introducere

> *“Caaaaaade, dă-te bă că cade!”*
> despre arbori, **Mihai Mărgineanu** (1969 - prezent)

Scopul temei constă în implementarea unui sistem de fișiere arborescent (conținând fițiere și directoare) ce poate fi manipulat prin intermediul comenzilor stil `bash`.

Cerință

Structura scheletului

Structura de bază pe care o veți folosi este `FileTree`:

```
struct FileTree {
    TreeNode* root;
};
```

Acesta conține elemente de tip `TreeNode`, ce pot fi atât fișiere cât și directoare:

```
struct TreeNode {
    TreeNode* parent;
    char* name;
    enum TreeNodeType type;
    void* content;
};
```

Fiecare dintre acestea are:

- director părinte
- nume
- tip: `FILE_NODE`/`FOLDER_NODE` (conform `enum`-ului `TreeNodeType`)
- conținut

Din moment ce conținutul diferă semnificativ în cazul în care nodul este un director, respectiv fișier, am ales să îl reprezentăm printr-o variabilă de tip `void*`. Aceasta va fi convertită în momentul folosirii într-unul din tipurile:

```
struct FileContent {
    char* text;
};
```

```
struct FolderContent {
    List* children;
};
```

Astfel, un fișier are drept conținut un șir de caractere ce nu conține spații, iar un director are o listă de copii, reprezentând resursele (fișiere/directoare) pe care le conține.

Nu în ultimul rând, această listă de resurse este reprezentată prin:

```
struct List {
    ListNode* head;
};
```

ce conține elemente de tip:

```
struct ListNode {
    TreeNode* info;
    ListNode* next;
};
```

Comenzi bash ce trebuie implementate - 80p

Recomandăm implementarea lor în aceeași ordine întrucat aceasta va fi și cea din checker

1. `touch <filename> [filecontent]` - **7p**
   - creează un fișier cu numele specificat în directorul curent.
   - dacă argumentul `filecontent` este specificat, acesta va deveni conținutul fișierului.
   - dacă există deja un fișier/director cu același nume, comanda nu va avea niciun efect.
   - ```
     $ touch a content_a

     $ touch b content_b

     $ ls
     b
     a
     ```
2. `ls [arg]` - **7p**
   - dacă este folosită fără niciun argument, comanda va lista resursele din directorul curent.
   - dacă argumentul este un director, comanda va lista resursele din cadrul acestuia.
   - dacă argumentul este un fișier, comanda va lista conținutul text al acestuia.
     ```

- dacă argumentul nu există, se va afișa eroarea `ls: cannot access '<arg>': No such file or directory`

```
$ touch a content_a

$ ls a
a: content_a

$ touch b content_b

$ ls b
b: content_b

$ touch c content_c

$ ls c
c: content_c

$ touch d content_d

$ ls
d
c
b
a
```

3. `mkdir <dirname>` - **7p**
   - creează un director cu numele dat în directorul curent.
   - dacă directorul deja există, va afișa eroarea: `mkdir: cannot create directory ‘<dirname>’: File exists`.

```
$ touch a content_a

$ ls
a

$ mkdir B

$ ls
B
a

$ touch c

$ ls
c
B
a
```

4. `cd <path>` - **7p**
   - comanda va porni din directorul curent și va urma calea descrisă de `<path>`, returnând directorul destinație.
   - `<path>` este de tipul `dir1/dir2/dir3/dir4/...`.
   - în `<path>` se poate face referire și la directorul părinte prin utilizarea `..`
   - dacă nu se poate ajunge la destinație, comanda va afișa eroarea: `cd: no such file or directory: <path>` și va returna directorul curent.
   - ```
     $ mkdir A

     $ cd A

     $ touch a

     $ ls
     a
     ```
5. `tree [path]` - **8p**
   - comanda va afișa ierarhia de fișiere/directoare, fie pornind din directorul curent, în cazul în care nu primește niciun argument, fie pornind de la un path dat.
   - de asemenea, comanda va afișa pe ultima linie numărul de directoare, respectiv fișiere vizitate, sub forma: `x directories, y files`.
   - în cazul în care `path`-ul nu poate fi parcurs sau se termină cu un fișier, se va afișa eroarea: `<path> [error opening dir]\n\n0 directories, 0 files\n`.
   - ```
     $ mkdir A

     $ cd A

     $ mkdir B

     $ cd B

     $ touch b

     $ cd ..

     $ mkdir C

     $ cd C

     $ touch c

     $ touch d

     $ cd ../..

     $ touch x content_x
     ```

```
$ tree
x
A
    C
        d
        c
    B
        b
3 directories, 4 files
```

6. pwd - **7p**
   - comanda va afișa calea absolută către directorul curent
   - după cum veți vedea și în exemplu, directorul de bază se numește “root”, însă nu va fi nevoie să tratați acest lucru explicit în implementare
   - ```
     $ mkdir A

     $ cd A

     $ pwd
     root/A
     ```
7. `rmdir <dirname>` - **7p**
   - comanda va șterge directorul specificat, numai în cazul în care el există și este gol.
   - dacă directorul nu există, se va afișa: `rmdir: failed to remove '<dirname>': No such file or directory`.
   - dacă directorul nu este gol, se va afișa: `rmdir: failed to remove '<dirname>': Directory not empty`.
   - daca resursa nu este un director, se va afișa: `rmdir: failed to remove '<dirname>': Not a directory`.
   - ```
     $ mkdir A

     $ mkdir B

     $ rmdir A

     $ ls
     B
     ```
8. `rm <filename>` - **7p**
   - comanda va șterge fișierul specificat.
   - dacă resursa nu este un fișier, se va afișa: `rm: cannot remove '<filename>': Is a directory`.
   - dacă fișierul nu există, se va afișa: `rm: failed to remove '<filename>': No such file or directory`

```
$ mkdir A

$ touch a

$ mkdir B

$ touch c

$ touch d

$ rm c

$ ls
d
B
a
A
```

9. `rmrec <resourcename>` - **7p**
   - comanda va șterge fișierul/directorul primit ca argument, indiferent dacă acesta este gol sau nu.
   - în cazul în care resursa nu există, se va afișa: `rmrec: failed to remove '<resourcename>': No such file or directory`.

```
$ mkdir A

$ mkdir B

$ cd B

$ touch b

$ cd ..

$ mkdir C

$ rmrec B

$ ls
C
A
```

10. `cp <source_path> <destination_path>` - **8p**
    - comanda va copia fișierul de la calea sursa la calea destinație.
    - dacă calea destinație e un nume de director existent, fișierul sursă va fi copiat în interiorul acestuia.
    - dacă calea destinație e un fișier existent, conținutul său va fi schimbat cu cel al fișierului sursă.
    - dacă calea destinație e un fișier care nu există dar ar putea fi creat, acesta va fi creat iar conținutul lui va fi cel al fișierului sursă.

- în cazul în care calea sursa este a unui director, se va afișa: `cp: -r not specified; omitting directory '<source_path>'`.
- în cazul în care calea destinație e o ierarhie de directoare ce nu poate fi accesată, se va afișa eroarea: `cp: failed to access '<destination_path>': Not a directory`.

```
$ mkdir A

$ cd A

$ touch a content_a

$ cp a ..

$ cd ..

$ ls
a
A
```

11. `mv <source_path> <destination_path>` - **8p**
    - comanda va muta fișierul/directorul de la calea sursă la calea destinație.
    - se aplică aceleași reguli ca cele de la comanda `cp`, cu excepția cazului în care sursa este un director, caz în care nu se va afișa o eroare, ci se va muta directorul, împreuna cu tot conținutul acestuia, la noua cale.

```
$ mkdir A

$ touch a

$ mv a A

$ tree
A
    a
1 directories, 1 files
```

Schelet cod + checker

Schelet cod + checker

Pentru ca testele să treacă și pe vmchecker, trebuie să folosiți flag-ul ”-std=c99” în cadrul regulii de build din Makefile

Menționați in README persoana cu care ați realizat tema. Tema trebuie trimisă de o singură persoană!

Depunctări

Eliberarea memoriei se va verifica folosind utilitarul Valgrind. O temă ce conține memory leaks pe un test va atrage după sine punctajul de 0p pe testul respectiv.

Nu copiați! Toate soluțiile vor fi verificate folosind o unealtă de detectare a plagiatului. În cazul detectării unui astfel de caz, atât plagiatorul cât și autorul original vor primi punctaj 0 pe toate temele! De aceea, vă sfătuim să nu vă lăsați rezolvări ale temelor pe calculatoare partajate (la laborator etc), pe mail/liste de discuții/grupuri etc.

Punctaj

- **80p** teste
- **10p** coding style
- **10p** README

FAQ

Q: Tema 3 se poate face în C++?

A: Nu.

Q: Se pot folosi variabile globale?

A: Nu.

Clarificări

Pentru întrebări sau nelămuriri legate de temă folosiți forumul temei [https://curs.upb.ro/2021/mod/forum/view.php?id=224229&forceview=1].

ATENȚIE să nu postați imagini cu părți din soluția voastră pe forumul pus la dispoziție sau orice alt canal public de comunicație. Dacă veți face acest lucru, vă asumați răspunderea dacă veți primi copiat pe temă.

sd-ca/teme/tema3-2022.txt · Last modified: 2022/05/25 17:37 by gabriel_danut.matei