Pythonbox

Cunostiinte evaluate:

- Utilizarea limbajului Python
- Intelegerea modului de functionare a liniei de comanda

Reguli:

1. Proiectul contine un fisier numit README.md care sa prezinta explicatii referitoare la modul de rezolvare al problemei.
2. Tema trebuie implementata in Python, utilizand **doar** functii din biblioteca **standard** de Python.

> **Atentie!**: Nu se pot utiliza biblioteci pentru executare de comenzi.

Pythonbox

Scopul acestui proiect este implementarea unui utilitar complet de executare a comenzilor de tip Linux bash, folosind limbajul Python.

Pentru a rezolva aceast proiect, se va realiza un program in Python care sa primeasca drept argument comanda pe care dorim sa o executam, urmata de parametrii sai. Programul executa comanda si apoi isi termina executia.

Exemplu ./main.py cp file folder

Comenzi acceptate

In continuare, vom defini comenzile suportate de utilitar, urmate de comoprtamentul caracteristic si parametrii acceptati. Pentru orice alta comanda sau format diferit, formatul va afisa mesajul **Invalid command** et va returna valoarea **-1**.

> Din motiv de incompatibilitate intre sistemul Linux si alte sisteme, codul de eroare afisat in terminalul Linux va fi egal cu 255, nu cu -1. Daca terminalul afiseaza 255 drept cod de eroare, solutia este corecta. Acest lucru se va produce in mod egal cu codurile de eroare specifice de mai jos, pentru fiecare din ele, terminalul va afisa un numar pozitiv.

In cazul in care comanda primita de catre utilitar se executa fara eroare, programul va returna valoarea 0. In caz contrar, va returna codul de eroare specific, mentionat in descrierea comenzii.

> Pentru a va ajuta sa intelegeti modul de functionare al fiecarei comenzi, am atasat fiecareia pagina din manualul de utilizare. Nu cerem implementarea tuturor parametrilor precizati in manual, doar a celor care sunt precizati in enuntul temei. Parametrii care au forma [param] sunt optionali.

Comenzile suportate de programul scris in Python sunt:

- **pwd** - Afiseaza calea absoluta a directorului curent doc

  Exemplu

  ```
  $ ./main.py pwd
  /home/pi/my_directories
  ```

- **echo [option] arguments** - Afiseaza argumentele consolei, urmate de o noua linie doc. In cazul unei erori se va returna valoarea -10 (se va afisa valoarea 246 in terminal).
  - **-n** - cu acest parametru nu vom adauga o linie noua la final

  Exemplu

  ```
  $ ./main.py echo a b c
  a b c
  $ ./main.py echo -n a b c
  a b c$
  ```

- **cat nume_fisiere** - Concateneaza continutul fisierelor si il afiseaza la iesirea standard doc. In caz de eroare va intoarce valoarea -20 (valoarea 236 va fi afisat in terminal)

  Exemplu

  ```
  $ ./main.py cat file1
  Text in file1
  $ ./main.py cat file2
  Text in file 2
  $ ./main.py cat file1 file2
  Text in file1
  Text in file 2
  ```

- **mkdir nume_directoare** - Creeaza directoarele trimise ca parametru, daca acestea nu exista. Daca operatiunea de nu poate fi efectuata, scriptul va returna valoarea -30 (valoarea 226 va fi afisata in terminal) doc

  Exemplu

```
./main.py mkdir my_drectory
./main.py mkdir my_drectory1 my_drectory2 my_drectory3
```

- **mv sursa destinatie** - Deplaseaza/Redenumeste fisierul/directorul sursa la cel destinatie doc. In caz de eroare se va intoarce va intoarce valoarea -40 (valoarea 216 va fi afisata in terminal)

  Exemplu

```
./main.py mv source destination
```

- **ln [optiune] sursa nume_link** - Creeaza un link simbolic nu muele nume_link catre fisierul sursa. Putem creea un link catre doar catre un fisier doc. In caz de eroare se va intoarce va intoarce valoarea -50 (valoarea 206 va fi afisata in terminal)
  - -s, –symbolic creeaza un link simbolic in locul unui hard link
    Exemplu

```
./main.py ln my_file my_file_link
./main.py ln -s my_file my_file_link3
```

- **rmdir nume_directoare** - Sterge directoarele goale pasate ca argumente doc. In caz de eroare se va intoarce va intoarce valoarea -60 (valoarea 196 va fi afisata in terminal)
  - -s, –symbolic creeaza un link simbolic in locul unui hard link

  Exemplu

```
./main.py rmdir my_empty_directory
./main.py rmdir my_empty_directory1 my_empty_directory2
```

- **rm [options] fichiere/directoare** - Sterge directoarele pasate ca argumente. Fara optiuni, nu poate sterge directoare. Daca primeste ca parametrii si fisiere, nu doar directoare, va sterge doar fisierele si va returna valoarea -70 (valoarea 186 va fi afisata in terminal) doc.
  - -r, -R, –recursive - sterge directoarele si continutul lor
  - -d, –dir - sterge directoarele fara continut

  Exemplu

```
./main.py rm my_file1 my_file2
./main.py rm -R my_directory
```

```
./main.py rm --dir my_empty_directory
```

- **ls [options] [director]** - Listeaza conitnutul directorului. Daca nu specificam niciun director, va lista continutul directorului curent; fara optiunea **-a/-all**, nu vom afisa fisierele/directoarele ascunse(cele ale caror nume nu incep cu '.'). Daca primim ca parametru calea catre un fisier, va afisa fisierul. Fiecare fisier/director va fi afisat pe o linie noua doc. In caz de eroare se va intoarce va intoarce valoarea -80 (valoarea 176 va fi afisata in terminal).
  - -a, -all - afiseaza si fisierele/directoarele ascunse
  - -R, –recursive - listeaza continutul fiecarui director din ierarhie. Pentru fisierele/directoarele care nu se gasesc direct in punctul citirii, va afisa calea absoluta, ex: output/test/file.

  Exemplu

```
 $ ./main.py ls
  directory1
  Directory2
  File1
  file2
$ ./main.py ls -a
  .
  ..
  directory1
  Directory2
  File1
  File2
$ ./main.py ls Directory2
  f1
  f2
```

- **cp [option] sursa destinatie** - Copiaza un fisier sau director de la sursa la destinatie. Daca nu mentionam numele destinatiei, fisierul va fi copiat cu numele sursei doc. In caz de eroare se va intoarce va intoarce valoarea -90 (valoarea 166 va fi afisata in terminal).
  - -R, -r, –recursive - copiaza in mod recursiv; se utilizeaza pentru a copia directoare cu tot continutul lor

  Exemplu

```
./main.py cp my_file my_directory
./main.py cp -r my_directory1 my_directory2
```

- **touch [options] fisier** -Actualizeaza data si ora de acces si modificare a fisierului la ora si data curente. daca fisierul nu exista, il creeaza la momentul executiei programului doc. In

caz de eroare se va intoarce va intoarce valoarea -100 (valoarea 156 va fi afisata in terminal).

- -a - schimba doar data si ora accesului
- -c, –no-creat - nu creeaza fisierul daca nu exista
- -m - schimba doar data si ora modificarii

Exemplu

```
./main.py touch my_file
./main.py touch -a --no-create my_file
```

- **chmod permisiuni fisier/director** - Schimba bitii de permisiune (rwx) al unui fisier/director doc. In caz de eroare se va intoarce va intoarce valoarea -25 (valoarea 231 va fi afisata in terminal). Permisiunile pot fi specificate in 2 moduri:
  i. numeric: un numar format din 3 cifre, fiecare reprezantand o valoare pe 3 biti: ex: 650
  ii. adaugand/stergand permisiuni specifice: pentru fiecare din cele 3 categorii (user, group others0 putem adauga sau sterge permisiuni. Categoriile sunt u - user, g - group, o - other, a - all. Formatul generic este: **u/g/o/a +/- r/w/x**.

Exemplu

```
./main.py chmod 570 file
./main.py chmod u+x file
./main.py chmod ug+rx file
./main.py chmod a-rx file
```

Bonus

- **grep [-i] regex nume_fisier** - Returneaza toate liniile fisierului care contin expresia regulata.
  - -i - Returneaza toate liniile fisierului care **nu** contin expresia regulata.

Exemplu

```
$ ./main.py grep '[0-9]+' File
  this line 99
  this line is another 7
```

- **ls**
  - -l - Afiseaza toate informatiile legate de fisiere

Exemplu

```
$ ./main.py ls -l
drwxr-xr-x alexandru staff 960 Feb 12 22:40 Desktop
-rw-r--r-- alexandru staff 372944 Nov 29  2020 Title Hello Wyliodrin STUDIO

# file_type (-, l - link, d - directory) properties user group size modifie
$ ./main.py ls -l Desktop
drwxr-xr-x alexandru staff 960 Feb 12 22:40 Desktop
```