Proiect

Introducere

Pentru rularea scriptului este necesar sa se creeze in directorul in care se afla scriptul fisierul text cu denumirea **rase_animale.txt** si cu urmatorul continut:

caine pomeranian

caine boxer

caine bulldog

caine poodle

caine beagle

caine yorkshire

caine dachshund

caine pug

pisica persiana

pisica bengaleza

pisica siameza

pisica sphynx

pisica chinchilla

pisica tiffanie

pisica tonkinese

Codul scriptului (comentat)

```
#!/bin/bash
```

#Intrucat in acest script se afla un meniu, scopul scriptului este mai variat
#Astfel, fiecare optiune a meniului a fost aleasa cu scopul introducerii a cat
#mai multor comenzi si cat mai variate.

#Prima optiune a meniului si prima parte a scriptului, de altfel, va verifica daca
#fisierul necesar pentru operatiile de la optiunile 3 si 4 exista. Fara acest fisier
#codul din script corespunzator acelor optiuni n-o sa functioneze.

#A doua optiune a meniului este bonus si reprezinta algoritmul lui fibonacci. L-am introdus
#aici pentru a incerca sa facem o trecere de la limbajele de programare pe care le stiam
#la comenzile specifice unui script bash. Este un algoritm cunoscut noua si ni s-a parut
#interesant ca pentru o intelegere mai buna a sintaxei sa ne folosim de cunostintele
#detinute. Pentru o mai buna intelegere, algoritmul lui Fibonacci afiseaza lista numerelor
#din sirul cu acelasi nume care are ca prime elemente pe 0 si 1, urmatoarele fiind
#reprezentate de suma celor 2 numere anterioare din sir. In cazul nostru, afiseaza primele
#n numere din sir, n fiind introdus de la tastatura.

#A treia optiune a meniului va afisa un alt meniu, care va avea doua optiuni, sa se
#afiseze doar rasele de caini sau doar cele de pisici, dar si numarul acestora.
#Aceasta parte a scriptului am facut-o pentru a scoate din fisierul text liniile care
#contin anumite cuvinte, dar si pentru a le numara pe acestea.

#A patra optiune arata o altfel de afisare a continutului fisierului text, inversand
#de fapt locurile numelui rasei cu cel al numelui animalului.

echo Alegeti ce doriti sa sa faceti: #aici ce vor afisa optiunile meniului

```
echo 1 - verificarea existentei fisierului text necesar rularii scriptului

echo 2 - pentru fibonacci

echo 3 - pentru lista animale

echo 4 - o altfel de afisare a tuturor raselor

echo Va rugam introduceti optiunea:

read menu #se va citi de la tastatura optiunea introdusa de utilizator


#incepand de aici se fac verificarile

if [ $menu -eq 1 ] #daca optiunea introdusa este egala cu 1, se va executa ce este in continutul urmatoarei structuri

then

        ls *_animale.txt #pentru optiunea 1, se va verifica existenta fisierului text, asa cum este prezentat si meniu

                #daca nu va aparea fisierul text ce contine rasele de caini si pisici, optiunea 2 si 3 din meniu nu va

                #fi disponibila


elif [ $menu -eq 2 ] #daca optiunea introdusa este egala cu 2, se va executa ce este in continutul urmatoarei structuri

then

        # se initializeaza valorile

        t1=0 # termenul 1

        t2=1 # termenul 2

        nextTerm=0 #urmatorul termen (o sa avem nevoie de el mai tarziu)

        counter=2 #avem deja primii 2 termeni daca numarul de termeni este mai mare decat 2, deci contorul este initializat cu 2

        # citirea numarului de termeni care se doreste a fi afisat

        echo Introduceti numarul de termeni:

        read n #se va citi de  la tastatura numarul termenilor a din sirul lui fibonacci care vor fi afisati
```

```
echo Seria Fibonacci pentru numarul de termeni introdus: #dupa aceasta va aparea sirul


if [ $n -gt 0 ] #verificare daca numarul de termeni ceruti este mai mare decat 0
then
        echo "$t1"  #va fi afisat primul termen daca numarul de termeni cerut este mai mare decat 0
fi


if [ $n -gt 1 ] #verificare daca numarul de termeni ceruti este mai mare decat 1
then
        echo "$t2" #va fi afisat al doilea termen daca numarul de termeni cerut este mai mare decat 1
fi


# inceperea prelucrarii celorlalti termeni
while [ $counter -lt $n ] #cat timp contorul este mai mic decat numarul termenilor care se doresc a fi afisati din
                                #sirul lui fibonacci se vor executa urmatoarele instructiuni
do


        nextTerm=$(($t1+$t2)) #termenul urmator o sa ia valoarea sumei dintre termenul 1 si termenul 2
        t1=$t2 # termenul 1 ia valoare termenului 2
        t2=$nextTerm #termenul 2 ia valoare urmatorului termen
        echo "$nextTerm " #se afiseaza valoare urmatorului termen
        counter=$(($counter+1)) #se incrementeza contorul
done
echo Sfarsit ":)" #se afiseaza pentru a marca sfarsitul sirului lui fibonacci pentru numarul de termeni cerut
```

```
elif [ $menu -eq 3 ] #daca optiunea introdusa este egala cu 3, se cor executa urmatoarele instructiuni

then

        echo Alegeti dintre urmatoarelec animale: #se afiseaza un scurt meniu secundar pentru a prezenta optiunile disponibile

        echo 1 - pentru a vedea rase de caini

        echo 2 - pentru a vedea rase de pisici

        read menu #se va citi de la tastatura optiunea introdusa pentru meniul secundar

        if [ $menu -eq 1 ] #daca optiunea introdusa este egala cu 1, se vor executa urmatoarele instructiuni

        then

                echo S-au gasit urmatoarele rase de caini:

                grep -i "caine" rase_animale.txt #se vor afisa toate liniile din fiserul text rase_animale.txt care contin

                                                #cuvantul -caine- fara a conta daca este scris in fisier cu majuscule sau nu

                echo S-au gasit rase de caini in numar de:

                grep -i "caine" rase_animale.txt | wc -l #se vor numara liniile care contin cuvantul -caine- din fierul

                                                        #rase_animale.txt, tot fara a tine cont daca cuvantul este scris in

                                                        #fisier cu majuscule sau nu

        elif [ $menu -eq 2 ] #daca optiunea introdusa este egala cu 2, se vor executa urmatoarele instructiuni

        then

                echo S-au gasit urmatoarele rase de pisici:

                grep -i "pisica" rase_animale.txt #se vor afisa toate liniile din fisierul text rase_animale.txt care contin

                                                #cuvantul -pisica- fara a conta daca este scris in fisier cu masucule sau nu

                echo S-au gasit rase de pisici in numar de:
```

```
        grep -i "pisica" rase_animale.txt | wc -l #se vor numara liniile care contin cuvantul -pisica- din fisierul
                                                  #rase_animale.txt, tot fara a tine cont daca cuvantul este scris
                                                  #in fiser cu majuscule sau nu
    else #altfel, daca se introduce orice alta optiune, diferita de 1 si 2, se va  executa urmatoarea instructiune
        echo Nu exista aceasta optiune! #se afiseaza ca nu exista o optiune de acest gen si se iese din script
    fi

elif [ $menu -eq 4 ] #daca optiunea introdusa este egala cu 4, se cor executa urmatoarele instructiuni
then
    sed -r 's/(\w+) (\w+)/\2, \1/' rase_animale.txt #aici se vor afisa cuvintele de pe fiecare linie, dar inversate si
                                                    #cu virgula intre cuvintele de pe aceeasi linie
    echo "" #l-am pus pentru a trece la linia urmatoare deoarece afisa localhost-ul in continuare ultimei linii afisate din
        #fiserul text

else #altfel, daca se introduce alta optiune, diferita de 1, 2, 3 sau 4, se va executa urmatoarea instructiune
    echo Nu exista aceasta optiune in meniu! #se va afisa ca nu exista o optiune de acest gen si se iese din script
fi
```

Istoricul generat dupa rularea scriptului

```
1  cd ./proiect
```

```
2  chmod a+x script.sh
3  ./script.sh
4  2
5  mc
6  ./script.sh
7  mc
8  ./script.sh
9  mc
10  ./script.sh
11  mc
12  ./script.sh
13  mc
14  ./script.sh
15  mc
16  ./script.sh
17  mc
18  ./script.sh
19  mc
20  ./script.sh
21  mc
22  ./script.sh
23  history > istoric.txt
```