Test practic la SO - varianta nr. 3

Realizați o aplicație formată din următoarele trei componente, definite în continuare, care vor fi dispuse conform ierarhiei de mai jos:

```
.
├── main.sh
├── legal_arts
│   └── iwashere.sh
    └── performances
        └── filegraffiti.c
```

1. Să se scrie un program C, numit "filegraffiti.c", conform specificației următoare:
Programul va primi două argumente în linia de comandă; în caz contrar se va termina cu codul de eroare 2.
Programul va considera primul argument ca fiind un nume de fișier și va înlocui ultimele N caractere din acest fișier cu caracterele cuvântului reprezentat de al doilea argument, unde N este lungimea acestuia. Operațiile pe fișier vor fi făcute cu API-ul POSIX și posibilele erori apărute în urma apelurilor de API vor fi abordate prin încheierea execuției programului cu un cod de terminare distinct pentru fiecare apel.
Se va afișa pe ieșirea stdout, folosind o funcție din biblioteca stdio, mesajul “CANVAS TOO SMALL\n”, dacă lungimea fișierului este mai mică decât N, iar altfel se va afișa, tot pe ieșirea stdout, mesajul “FILE VANDALIZED: <filepath>\n”, unde <filepath> este numele fișierului modificat. În final, programul își va încheia execuția cu codul de terminare 0. (Atenție cu ce argumente apelați programul; vă recomand să rulați programul sub un user ce nu are privilegii root, pentru a nu corupe din greșeală fișiere esențiale.)

2. Să se scrie un script bash, numit "iwashere.sh", conform specificației următoare:
Scriptul va primi un argument în linia de comandă; în caz contrar se va termina cu codul de eroare 2.
Scriptul va considera acel argument ca fiind un nume de fișier. Dacă fișierul este de tip obișnuit, scriptul va apela executabilul “filegraffiti” (obținut prin compilarea sursei “filegraffiti.c”), transmițându-i acestuia ca argumente în linia de comandă, numele fișierului și numele utilizatorului ce rulează scriptul curent.
În schimb, dacă numele de fișier primit este un director, scriptul va efectua următoarea procesare a conținutului acelui director: va parcurge, folosind o structură repetitivă, toate intrările directe din acel director și, pentru fiecare intrare, **scriptul se va apela recursiv pe sine însuși**, cu acea intrare ca parametru în linia de comandă. (Atenție: așadar, aici trebuie să implementați o recursie explicită!)

3. Să se scrie un alt script bash, numit "main.sh", conform specificației următoare:
Scriptul va primi un argument în linia de comandă și va face următoarele verificări, în ordinea următoare:
i) va verifica că în directorul în care se află "main.sh", există un subdirector numit "legal_arts" ce conține scriptul "iwashere.sh", iar în caz negativ va afișa pe ieșirea stderr un mesaj de eroare adecvat și se va termina cu codul 1. În plus, în cazul în care nu are drept de execuție pentru "iwashere.sh", o va adăuga.
ii) va verifica că în directorul în care se află "iwashere.sh", există un subdirector numit "performances" ce conține programul "filegraffiti.c" și că are permisiunea de citire a acestuia, iar în caz negativ va afișa pe ieșirea stderr un mesaj de eroare adecvat și se va termina cu codul 2;
iii) va verifica că în subdirectorul "performances" se află și un fișier numit "filegraffiti" (executabilul obținut din sursa "filegraffiti.c"), iar în caz negativ va apela compilatorul gcc pentru a-l produce și, doar dacă vor fi erori la compilare, va afișa pe ieșirea stderr un mesaj de eroare adecvat și se va termina cu codul 3;
iv) va verifica dacă argumentul primit reprezintă un director existent și asupra căruia are permisiune de citire, iar în caz negativ va afișa pe ieșirea stderr un mesaj de eroare adecvat și se va termina cu codul 4.

Apoi scriptul "main.sh" va invoca scriptul "iwashere.sh", transmițându-i în linia de comandă, argumentul pe care l-a primit el în linia de comandă. Invocarea scriptului "iwashere.sh" se va face printr-o comandă compusă de tip pipeline, care să proceseze outputul produs prin execuția scriptului "iwashere.sh" în maniera următoare: se vor sorta rândurile din output în ordine invers lexicografică și se vor elimina cele duplicat, după care se va înlocui fiecare apariție a caracterului “/” (forward slash) cu șirul vid (în alte cuvinte, se vor elimina toate aparițiile caracterului “/”).
La finalul procesării pipeline-ului, scriptul "main.sh" se va termina cu codul de terminare 0.

Barem de corectare

Observații:
- programul C și cele două scripturi trebuie plasate într-o ierarhie de directoare conform celor descrise în enunțul problemei (și apelate în mod corespunzător poziției lor în ierarhia respectivă).
- conform specificației date, programul C poate folosi biblioteca standard de C doar pentru afișarea rezultatelor; interacțiunea cu fișierul (poziționarea în fișier, suprascrierea conținutului, etc.) se va face folosind doar API-ul POSIX.
- pentru implementarea parcurgerii recursive se va respecta specificația dată pentru scriptul "iwashere.sh".
- **dacă nu respectați toate condițiile precedente** (de exemplu, dacă plasați vreunul dintre cele două fișiere apelate de “main.sh” într-un alt director, sau dacă folosiți comenzi precum find sau ls -R pentru parcurgerea recursivă), atunci vom evalua corectitudinea rezolvării, dar **punctajul total acordat va fi înjumătățit.**
- fiecare punctaj din barem se acordă integral, sau deloc.

1. Baremul pentru programul "filegraffiti.c":
a) 2p - implementarea corectă, conform specificației date, a sesiunii de lucru cu fișierul dat ca argument pentru realizarea modificării conținutului acestuia, în maniera specificată.
b) 1p - afișarea rezultatului pe ieșirea stdout, conform specificației date.
c) 1p - tratarea tuturor erorilor posibile, conform specificației date.
d) 1p - programul se compilează fără erori și fără warning-uri.
Total: 5p

2. Baremul pentru scriptul "iwashere.sh":
a) 1p - luarea deciziilor corecte de procesare, după cum argumentul primit este un director sau un fișier obișnuit (regular), conform specificației date.
b) 1p - invocarea corectă a programului executabil "filegrafitti", dacă argumentul primit este un fișier normal, conform specificației date.
c) 2p - iar dacă argumentul primit este un director, parcurgerea corectă a intrărilor directe din directorul respectiv și apelarea recursivă a scriptului pentru fiecare intrare din acesta, conform specificației date.
d) 1p - scriptul nu are erori de sintaxă la execuție.
Total: 5p

3. Baremul pentru scriptul "main.sh":
a) 2p - implementarea corectă, conform specificației date, a verificării descrise la punctul i) din specificație.
b) 1p - implementarea corectă, conform specificației date, a verificării descrise la punctul ii) din specificație.
c) 1p - implementarea corectă, conform specificației date, a verificării descrise la punctul iii) din specificație.
d) 1p - implementarea corectă, conform specificației date, a verificării descrise la punctul iv) din specificație.
e) 1p - invocarea corectă a scriptului "iwashere.sh", în cadrul pipeline-ului, conform specificației date.
f) 1p - apelul comenzii potrivite, în cadrul pipeline-ului, pentru sortarea rândurilor și eliminarea celor duplicat.
g) 1p - apelul comenzii potrivite, în cadrul pipeline-ului, pentru a elimina toate aparițiile caracterului “/” (forward slash).
h) 1p - scrierea corectă a întregului pipeline pentru procesarea descrisă în specificația dată.
i) 1p - scriptul nu are erori de sintaxă la execuție.
Total: 10p