Test practic la SO - varianta nr. 3

Realizați o aplicație formată din următoarele trei programe cooperante, plus un script de execuție a lor, care vor fi plasate conform ierarhiei de directoare de mai jos:

```
.
├── starter.sh
├── master
│   └── supervisor.c
└── slaves
    └── worker1.c
    └── worker2.c
```

1. Programul "supervisor.c" va implementa funcționalitatea descrisă în specificația următoare:
Programul va primi la linia de comandă calea (absolută sau relativă) spre un fișier existent pe disc, numit "input.txt", ce conține diverse caractere alfanumerice, și va face următoarele operații:
i) În funcția principală a programului, va crea o mapare ne-persistentă cu numele "comunicare_decriptie" (i.e., un obiect de memorie partajată).
ii) Și, tot în funcția principală a programului, va inițializa un canal anonim, pentru a putea transmite date către procesul creat la iii).
iii) De asemenea, tot în funcția main, va crea un proces copil, în care va lansa în execuție programul **worker1** folosind o primitivă exec adecvată; comunicarea cu procesul **worker1** se va realiza prin intermediul canalului anonim inițializat la ii).
iv) Într-o funcție separată, apelată din funcția main, programul va citi fișierul de intrare și va trimite către procesul **worker1,** prin intermediul canalului anonim, toate caracterele ce sunt litere mari sau litere mici, în ordinea citirii lor din fișierul de intrare (astfel se elimină cifrele și alte tipuri de caractere citite din textul de intrare).
v) Într-o altă funcție separată, apelată din funcția main, programul va citi din maparea ne-persistentă cu nume ("comunicare_decriptie") rezultatul final calculat de worker2, și îl va afișa pe ecran. *Atenție***:** înainte de a afișa conținutul mapării, trebuie să vă asigurați că acesta este diferit de 0.

2. Programul "worker1.c" va implementa funcționalitatea descrisă în specificația următoare:
i) În funcția principală a programului, acesta va crea un canal fifo cu numele "send_data" pe care-l va folosi pentru comunicarea cu procesul **worker2**.
ii) Într-o funcție separată, apelată din funcția main, programul va face următoarea procesare: pentru fiecare caracter primit prin canalul anonim de la **supervisor** va efectua următoarea operație: (caracter - 'A' + 14) % 26 + 'A', dacă caracterul citit este literă mare, sau (caracter - 'a' + 14) % 26 + 'a', dacă caracterul este literă mică; valoarea astfel obținută este transmisă către **worker2** prin canalul fifo creat anterior. *Atenție***:** procesul **worker1** nu va stoca în memorie toate caracterele primite, ci pur și simplu le va trimite mai departe, modificate, către procesul **worker2**.

3. Programul "worker2.c" va implementa funcționalitatea descrisă în specificația următoare:
i) i) În funcția principală a programului, acesta va face inițializările necesare pentru a putea comunica cu procesul **supervisor** (prin obiectul de memorie partajată "comunicare_decriptie").
ii) De asemenea, tot în funcția main, va face inițializările necesare pentru a putea comunica cu procesul **worker1** (prin canalul fifo "send_data").
iii) Într-o funcție separată, apelată din funcția main, programul va număra câte vocale și, respectiv, câte consoane primește de la **worker1** prin canalul fifo; apoi va trimite perechea de valori astfel calculate mai departe, către **supervisor**, prin intermediul obiectului de memorie partajată ("comunicare_decriptie").

4. Scriptul "starter.sh" va implementa funcționalitatea descrisă în specificația următoare:
i) Scriptul va porni mai întâi execuția programului **worker2** (din subdirectorul corespunzător) în *background.*
ii) După o pauză de 3 secunde, va porni și execuția programului **supervisor**, cu un argument în linia de comandă: calea (absolută sau relativă) către fișierul de intrare "input.txt" (pe care-l veți crea anterior, în orice editor de texte doriți, cu formatul specificat în enunț).

Recomandare:

Mai întâi, desenați de mână pe o foaie de hârtie o schemă cu "arhitectura aplicației": ierarhia de procese, mijloacele de comunicație între procese și sensul de transmitere a informației prin aceste mijloace de comunicație (precum ați văzut în exemplele de diagrame realizate de mână, atașate la unele exerciții de laborator).
Astfel vă veți clarifica mai bine cerințele specificate în enunțul problemei și, de asemenea, vă va ajuta la partea de implementare!

Submitere:

La finalul testului, pentru a submite rezolvarea dvs. printr-un formular Google ce vă va fi indicat, veți face o arhivă (în format .zip sau .tar / .tar.gz) care să conțină cele 3 programe sursă C (plus eventuale fișiere header .h proprii, în cazul în care veți defini și folosi astfel de fișiere), scriptul starter.sh și fișierul de intrare input.txt, **cu numele lor și poziția în ierarhia de directoare exact precum au fost specificate** în enunțul de mai sus. Iar arhiva o veți denumi în felul următor:

NumePrenume_TP2.zip (sau .tar sau .tar.gz)

Barem de corectare

Observații:
- programele și scriptul trebuie plasate într-o ierarhie de directoare conform celor descrise în enunțul problemei (și apelate în mod corespunzător poziției lor în ierarhia respectivă); de asemenea, conținutul arhivei finale va păstra ierarhia respectivă de directoare și fișiere sursă.
- pentru implementarea comunicațiilor între cele 3 procese cooperante se vor utiliza exact metodele de comunicare descrise în specificații.
- **dacă nu respectați toate condițiile precedente** (de exemplu, dacă folosiți, pentru vreuna dintre comunicații, un alt mijloc de comunicație decât cel specificat, sau dacă plasați vreunul dintre programele pornite din “starter.sh” într-un alt director decât cel specificat, 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 "supervisor.c":
a) 0.5p - implementarea corectă a inițializărilor de comunicații descrise la punctul i) din specificația dată.
b) 0.5p - implementarea corectă a inițializărilor de comunicații descrise la punctul ii) din specificația dată.
c) 1p - implementarea corectă a operațiilor fork și exec descrise la punctul iii) din specificația dată.
d) 1p - implementarea corectă a procesărilor de date descrise la punctul iv) din specificația dată.
e) 0.5p - implementarea corectă a trimiterii de date descrise la punctul iv) din specificația dată.
f) 1p - implementarea corectă a comunicației prin *shared-memory* și afișarea pe ecran a rezultatului citit, conform celor descrise la punctul v) din specificația dată.
Notă: se va calcula fracția procentuală de îndeplinire a criteriilor de mai sus, iar cele trei criterii de mai jos vor fi notate prin ponderare cu acest procentaj: F = total puncte acordate pentru criteriile a)-f) / 4.5p
g) F * 1p - tratarea cazurilor de eroare la TOATE apelurile de sistem din program.
h) F * 1p - programul se compilează fără erori și fără warning-uri.
i) F * 0.5p - scrierea codului într-o formă indentată, plăcută ochiului, i.e. "readable code" (de exemplu, evitați copy-paste în mcedit).
Total: 7p

2. Baremul pentru programul "worker1.c":
a) 0.5p - implementarea corectă a inițializărilor de comunicații descrise la punctul i) din specificația dată.
b) 0.5p - implementarea corectă a comunicației prin *canal anonim* descrise la punctul ii) din specificația dată.
c) 1p - implementarea corectă a procesărilor de date descrise la punctul ii) din specificația dată.
d) 1p - implementarea corectă a comunicației prin *canal fifo* descrise la punctul ii) din specificația dată.
Notă: se va calcula fracția procentuală de îndeplinire a criteriilor de mai sus, iar cele trei criterii de mai jos vor fi notate prin ponderare cu acest procentaj: F = total puncte acordate pentru criteriile a)-d) / 3p
e) F * 1p - tratarea cazurilor de eroare la TOATE apelurile de sistem din program.
f) F * 1p - programul se compilează fără erori și fără warning-uri.
g) F * 0.5 - scrierea codului într-o formă indentată, plăcută ochiului, i.e. "readable code" (de exemplu, evitați copy-paste în mcedit).
Total: 5.5p

3. Baremul pentru programul "worker2.c":
a) 0.5p - implementarea corectă a inițializărilor de comunicații descrise la punctul i) din specificația dată.
b) 0.5p - implementarea corectă a inițializărilor de comunicații descrise la punctul ii) din specificația dată.
c) 1p - implementarea corectă a comunicației prin *canal fifo* descrise la punctul iii) din specificația dată.
d) 1p - implementarea corectă a procesărilor de date descrise la punctul iii) din specificația dată.
e) 1p - implementarea corectă a comunicației prin *shared-memory* descrise la punctul iii) din specificația dată.
Notă: se va calcula fracția procentuală de îndeplinire a criteriilor de mai sus, iar cele trei criterii de mai jos vor fi notate prin ponderare cu acest procentaj: F = total puncte acordate pentru criteriile a)-e) / 4p
f) F * 1p - tratarea cazurilor de eroare la TOATE apelurile de sistem din program.
g) F * 1p - programul se compilează fără erori și fără warning-uri.
h) F * 0.5 - scrierea codului într-o formă indentată, plăcută ochiului, i.e. "readable code" (de exemplu, evitați copy-paste în mcedit).
Total: 6.5p

4. Baremul pentru scriptul "starter.sh":
a) 2 x 0.5p - implementarea corectă a apelării celor două programe în modul descris în specificația dată.
Total: 1p

Total general 1. - 4. : 20p