Test practic la SO - varianta nr. 1

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
├── coordonator
│   └── supervisor.c
└── subordinates
    └── worker1.c
    └── worker2.c
```

1. Programul "supervisor.c" va implementa funcționalitatea descrisă în specificația următoare:
Programul va primi un argument în linia de comandă, ce reprezintă calea (absolută sau relativă) către un fișier existent pe disc, numit "operatii.txt". Acest fișier conține un număr variabil de operații de forma "*termen operator termen*", fiecare operație pe câte o linie de text. Termenii sunt numere întregi în format textual, iar operatorii sunt caracterele: + (adunare), - (scădere), * (înmulțire) sau / (împărțire întreagă).
i) În funcția principală a programului, acesta va testa existența unui argument primit în linia de comandă și va face inițializările necesare pentru a putea trimite informații procesului **worker1** printr-un canal fifo și, respectiv, pentru a putea comunica cu procesul **worker2** printr-o mapare ne-persistentă cu nume (i.e., un obiect de memorie partajată).
ii) Într-o funcție separată, apelată din funcția main, programul va citi pe rând, una câte una, fiecare linie din acel fișier "operatii.txt" și, după parsarea liniei respective, va transmite tripleta de doi operanzi și un operator către procesul **worker1** printr-un canal fifo (vă recomand să folosiți reprezentarea binară a numerelor întregi pentru a avea mesaje de lungime constantă).
iii) Într-o altă funcție separată, apelată din funcția main, programul va citi cele două numere (rezultate) transmise lui de către procesul **worker2** prin intermediul unei mapări ne-persistente cu nume și calculează suma acestor numere, afișând-o la final pe ecran.

2. Programul "worker1.c" va implementa funcționalitatea descrisă în specificația următoare:
i) În funcția principală a programului, acesta va crea un proces fiu, iar în fiul creat va starta, printr-un apel exec adecvat, programul **worker2**.
ii) De asemenea, tot în funcția main, va face inițializările necesare pentru a putea primi informații de la procesul **supervisor** printr-un canal fifo și, respectiv, pentru a putea trimite informații procesului **worker2** printr-un canal anonim.
iii) Într-o funcție separată, apelată din funcția main, programul va citi (vă recomand să folosiți reprezentarea binară a numerelor întregi pentru a avea mesaje de lungime constantă) din acel canal fifo, rând pe rând, câte o operație transmisă lui de către procesul **supervisor** și va calcula rezultatul acelei operații. Fiecare rezultat astfel obținut va fi transmis mai departe către procesul **worker2,** prin acel canal anonim.

3. Programul "worker2.c" va implementa funcționalitatea descrisă în specificația următoare:
i) În funcția principală a programului, va face inițializările necesare pentru a putea comunica cu procesul **supervisor** printr-o mapare ne-persistentă cu nume (i.e., un obiect de memorie partajată).
ii) Într-o funcție separată, apelată din funcția main, programul citește, rând pe rând, rezultatele transmise lui de către procesul **worker1** (prin canalul anonim creat de acesta) și calculează, fără a stoca în memoria proprie întreaga secvență de rezultate primite, maximul numerelor *pare* și, respectiv, minimul numerelor *impare*, din secvența de rezultate primite. Apoi transmite cele două numere calculate (i.e., maximul și minimul) către procesul **supervisor**, prin acel obiect de memorie partajată.

4. Scriptul "starter.sh" va implementa funcționalitatea descrisă în specificația următoare:
i) Scriptul va porni mai întâi execuția programului **worker1** (din subdirectorul corespunzător) în *background*.
ii) După o pauză de 2 secunde, va porni și execuția programului **supervisor**, cu un argument în linia de comandă: calea (absolută sau relativă) către fișierul de intrare "operatii.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 operații.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) 1p - implementarea corectă a inițializărilor de comunicații descrise la punctul i) din specificația dată.
b) 1p - implementarea corectă a procesărilor de date descrise la punctul ii) din specificația dată.
c) 1p - implementarea corectă a comunicației prin *canal fifo* descrise la punctul ii) din specificația dată.
d) 0.5p - 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) / 4.5p
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 * 0.5p - 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) 1p - implementarea corectă a operațiilor fork și exec descrise la punctul i) din specificația dată.
b) 1p - implementarea corectă a inițializărilor de comunicații descrise la punctul ii) din specificația dată.
c) 1p - implementarea corectă a procesărilor de date descrise la punctul iii) din specificația dată.
d) 1p - implementarea corectă a comunicației prin *canal fifo* descrise la punctul iii) din specificația dată.
e) 0.5p - implementarea corectă a comunicației prin *canal anonim* 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) / 4.5p
f) F * 1p - tratarea cazurilor de eroare la TOATE apelurile de sistem din program.
d) 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: 7p

3. Baremul pentru programul "worker2.c":
a) 0.5p - implementarea corectă a inițializărilor de comunicații descrise la punctul i) din specificația dată.
b) 1p - implementarea corectă a procesărilor de date descrise la punctul ii) din specificația dată.
c) 0.5p - implementarea corectă a comunicației prin *canal anonim* descrise la punctul ii) din specificația dată.
d) 0.5p - implementarea corectă a comunicației prin *shared-memory* 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) / 2.5p
f) F * 1p - tratarea cazurilor de eroare la TOATE apelurile de sistem din program.
d) 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: 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