

1. Hozzon létre azonos szülőfolyamatból két gyerekfolyamatot. A szülő írja ki a saját és a két gyerek PID-jét, a gyerekek pedig a saját és a szülőjük PID-jét úgy, hogy beazonosítható legyen, mit melyik folyamat ír ki.

```
Szülő: Saját PID = 5393
Szülő: Gyerek1 PID = 5394
Szülő: Gyerek2 PID = 5395
Gyerek1: Saját PID = 5394
Gyerek1: Szülő PID = 5393
Gyerek2: Saját PID = 5395
Gyerek2: Szülő PID = 5393
```

2. Készítse el az 1. feladatot úgy, hogy a gyerekfolyamatok kódját ne írja meg kétszer. (Tipp: A gyerekfolyamatok ugyanazt csinálják, csak a kiíráskor az egyik azt írja a sorok elejére, hogy „Gyerek1:”, a másik pedig, hogy „Gyerek2:”, ez pedig átadható paraméterként egy függvénynek).

3. Módosítsa úgy a 2. feladat megoldását, hogy ne keletkezzenek zombi folyamatok!

4. Hozzon létre egy gyerekfolyamatot, majd abból egy újabb gyerekfolyamatot. A szülő írja ki a saját és a gyerek PID-jét, a gyerekek pedig a saját és a szülőjük PID-jét úgy, hogy beazonosítható legyen, mit melyik folyamat ír ki.

```
Szülő: Saját PID = 6518
Szülő: Gyerek1 PID = 6519
Gyerek1: Saját PID = 6519
Gyerek1: Szülő PID = 1
Gyerek2: Saját PID = 6520
Gyerek2: Szülő PID = 6519
```

5. Hozzon létre egy gyerekfolyamatot, majd abból egy újabb gyerekfolyamatot. Mindhárom folyamat írja ki a saját és a szülője PID-jét úgy, hogy beazonosítható legyen, mit melyik folyamat ír ki. Mindhárom folyamat ugyanazt a kódot futtassa, ne írja le ugyanazt a kódot háromszor!

```
Szülő: Saját PID = 6764
Szülő: Szülő PID = 5322
Gyerek1: Saját PID = 6765
Gyerek1: Szülő PID = 1
Gyerek2: Saját PID = 6766
Gyerek2: Szülő PID = 6765
```

6. Módosítsa úgy az 5. feladat megoldását, hogy ne keletkezzenek zombi folyamatok!

7. Készítsen programot, melynek egyetlen paramétere van. A programban hozzon létre azonos szülőfolyamatból annyi gyerekfolyamatot, amennyi a paraméter értéke. Minden gyerek írja ki a saját sorszámát és PID-jét.

8. Készítsen programot, melynek szétválasztása előtt létrehozza x változót 10-es kezdőértékkel, majd hozzon létre azonos szülőfolyamatból két gyerekfolyamatot. Mindhárom folyamat hozzon létre egy-egy y változót. Ezután mindhárom folyamat írja ki x értékét, majd növelje azt 1-gyel és ismét írja ki. Végül mindhárom folyamat írja ki x és y memóriacímét. (Tipp: Ha a memóriacímét előjel nélküli egészként szeretné kiírni, használja a %u formátumkódot.)

```
Szülő kezdetben: x=10
Szülő végén: x=11
Szülőben x címe: 2212727508
Szülőben y címe: 2212727504
Gyerek2 kezdetben: x=10
Gyerek1 kezdetben: x=10
Gyerek2 végén: x=11
Gyerek1 végén: x=11
Gyerek2 x címe: 2212727508
Gyerek1 x címe: 2212727508
Gyerek2 y címe: 2212727500
Gyerek1 y címe: 2212727496
```

9. Készítsen programot, melynek egyetlen paramétere van. A programban hozzon létre azonos szülőfolyamatból annyi gyerekfolyamatot, amennyi a paraméter értéke. Az egyik gyerek írjon ki 10 sorba 10 db 'A' betűt. A másik 10 db 'B'-t, a harmadik 10 db 'C'-t, és így tovább.

10. Hozzon létre 2 gyerekfolyamatot, amelyek egyenként 50 sorba soronként 1 db 1-100 közötti véletlenszámot írnak ki a képernyőre. Minden véletlenszám után az azt kiíró folyamat PID-je is kerüljön kiírásra! A szülő várja meg a gyerekfolyamatok befejeződését! A két gyerek ne ugyanazt a véletlen számsort írja ki!

11. Hozzon létre azonos szülőfolyamatból két gyerekfolyamatot. A szülő írja ki a két gyerek PID-jét, várjon 5 másodpercig, végül hajtson végre 3 wait-et, és írja ki azok visszatérési értékét. A két gyerek 1 illetve 2 másodperc várakozást követően írja ki a saját és a szülője PID-jét, majd termináljon.

12. Módosítsa a 11. feladatban leírt programot úgy, hogy az 1 másodpercet várakozó gyerek várjon további 10 másodpercet a PID-ek kiírása és a terminálás között!

13. Hozzon létre azonos szülőfolyamatból két gyerekfolyamatot.

- A szülő írja ki a két gyerek PID-jét, majd hajtson végre 4 waitpid-et, és írja ki azok visszatérési értékét, valamint, hogy mire várakoztak.
 - Az első waitpid a 2. gyerek terminálására várakozzon,
 - a második bármely folyamat terminálására,
 - a harmadik a szülő szülőjének terminálására,
 - a negyedik ismét bármelyik folyamat terminálására.
- Az 1. gyerek írja ki a saját és a szülője PID-jét, majd termináljon.
- A 2. gyerek 5 másodperc várakozást követően írja ki a saját és a szülője PID-jét, majd termináljon.

14. Hozzon létre a szülőfolyamatból egy gyerekfolyamatot (gy1), majd a gyerekfolyamatból egy másik gyerekfolyamatot (gy2).

- A szülő hajtson végre két waitpid-et, melyek bármely gyerek terminálására várnak, és írja ki azok visszatérési értékét, valamint, hogy mire vártak.
- Az 1. gyerek (gy1) írja ki a saját és a szülője PID-jét, majd termináljon.
- A 2. gyerek (gy2) 5 másodperc várakozást követően írja ki a saját és a szülője PID-jét, majd termináljon.

15. Módosítsa a 14. feladat megoldását úgy, hogy az 1. gyerek (gy1) mielőtt terminálna, várja meg a 2. gyerek (gy2) terminálását.

16. Készítsen programot, melynek egyetlen paramétere van. A programban hozzon létre azonos szülőfolyamatból annyi gyerekfolyamatot, amennyi a paraméter értéke. Minden gyerek írja ki a saját sorszámát és PID-jét. A szülő várja meg az összes gyerek terminálását, és írja is ki minden esetben, hogy melyik gyerek (pid) állt le!

```
klettner@szamrend:~/opsys/feladatok4$ a.out 5
1. gyerek. PID=16492
2. gyerek. PID=16493
4. gyerek. PID=16495
5. gyerek. PID=16496
16492 leállt!
16493 leállt!
3. gyerek. PID=16494
16495 leállt!
16496 leállt!
16494 leállt!
```

17. Módosítsa a 16. feladatban leírt programot úgy, hogy egyidőben legfeljebb 5 folyamat (1 szülő és 4 gyerek) fusson! A gyerekek pedig várjanak 1 másodpercet a PID kiírása és a terminálás között! (Tehát pl. az 5. gyereket addig ne indítsa el, amíg az első 4 gyerek valamelyike nem terminál, és így tovább...)

18. Módosítsa a 17. feladatban leírt programot úgy, hogy a gyerekek véletlenszerű ideig (1-5 másodperc) várjanak a PID kiírása és a terminálás között!

(Ötlet: Ahhoz, hogy a gyerekfolyamatok ne ugyanazokat a véletlen értékeket állítsák elő, inicializálja a véletlenszám generátort a rendszeridő és a gyerekfolyamat pid-jének együttes használatával előállított értékkel.)

19. Készítse el „pid” programot, mely, ha nem kap paramétert, akkor kiírja a saját pid-jét egyszer, ha pedig kap egy paramétert, annyiszor írja ki a saját pid-jét, amennyi a paraméter értéke, végül annyi másodpercet várjon, ahányszor kiírta a pid-jét!

Ezt követően készítsen egy másik programot, mely szintén kiírja saját PID-jét, majd lecseréli magát ./pid programra 3-as értékű paraméter átadásával! Próbálja kiírni egy, a lecserélést követő utasításban az „Ide sohasem jut el!” szöveget. Figyelje meg, kiíródik-e ez a szöveg.

20. Készítsen programot, mely kiírja a saját pid-jét, majd lefuttatja az előző feladatban létrehozott ./pid programot 2-es értékű paraméter átadásával. Végül lefuttatja az „ls -al” unix parancsot!

21. Indítson egy programból pontosan 3 folyamatot, melyek unix parancsok használatával különböző feladatokat hajtanak végre! Az egyik hozzon létre egy fájlt, a másik listázza ki az aktuális könyvtár tartalmát, a harmadik pedig törölje a létrehozott fájlt. Ügyeljen a folyamatok szinkronizációjára. (Segítségül unix parancsok: „touch file”, „rm file” „ls”.)

22. Készítsen saját shell-t! A saját shell-ből való kilépés az exit paranccsal történjen.

```
klettner@szamrend:~/opsys/feladatok4$ a.out
shell> ls
10.c  13.c  16.c  19.c  21.c  3.c   6.c   8.c   pid
11.c  14.c  17.c  1.c   22.c  4.c   7a.c  9.c   pid.c
12.c  15.c  18.c  20.c  2.c   5.c   7b.c  a.out
shell> gcc 22.c
shell> exit
klettner@szamrend:~/opsys/feladatok4$ █
```