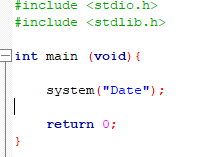
***Operációs Rendszerek BSC  
5***. gyakorlat  
2021.március 10.

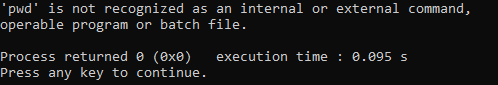
**Készítette:**Kércsi Bence  
Programtervező Informatikus  
ILVIYV

1. A system() rendszerhívással hajtson végre létező és nem létező parancsot, és vizsgálja a visszatérési érteket! Mentés: *neptunkodgyak1.c*

A kód:  




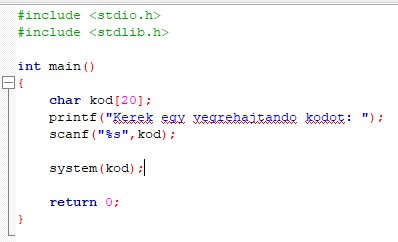
Ezzel a kóddal be tudnék új időt állítani.

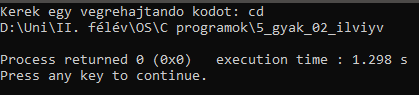
Ezzel szemben, például a pwd-vel:  


// Végül egy cd-s változat került feltöltésre

1. Írjon programot, amely billentyűzetről bekér Unix parancsokat és végrehajtja őket, majd kiírja a szabványos kimenetre. (pl.: amit bekér: date, pwd, who etc.; kilépés: CTRL-\)

A kód:



Eredmény:  


1. Készítsen egy parent.c és a child.c programokat. A parent.c elindít egy gyermek processzt, ami különbözik a szülőtől. A szülő megvárja a gyermek lefutását. A gyermek szöveget ír a szabványos kimenetre (5-ször) (pl. a hallgató neve és a neptunkód)!
2. 4. A fork() rendszerhívással hozzon létre egy gyerek processzt-t és abban hívjon meg egy exec

családbeli rendszerhívást (pl. execlp). A szülő várja meg a gyerek futását!

5. A fork() rendszerhívással hozzon létre gyerekeket, várja meg és vizsgálja a befejeződési állapotokat (gyerekben: exit, abort, nullával való osztás)!

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

int main()

{

pid\_t pid=fork();

if(pid<0){

printf("Fork hiba");

exit(-1);

}

else if (pid==0){

execl("./child","child",(char \*)NULL);

}

int status;

waitpid(pid,&status,0);

if(WIFEXITED(status)){

int exitStatus=WEXITSTATUS(status);

printf("A kilepes erteke: %d\n",exitStatus);

}

return 0;

}

#include <stdio.h>

#include <stdlib.h>

int main()

{

for (int i=0;i<5;i++) {

printf("Hello word!\n");

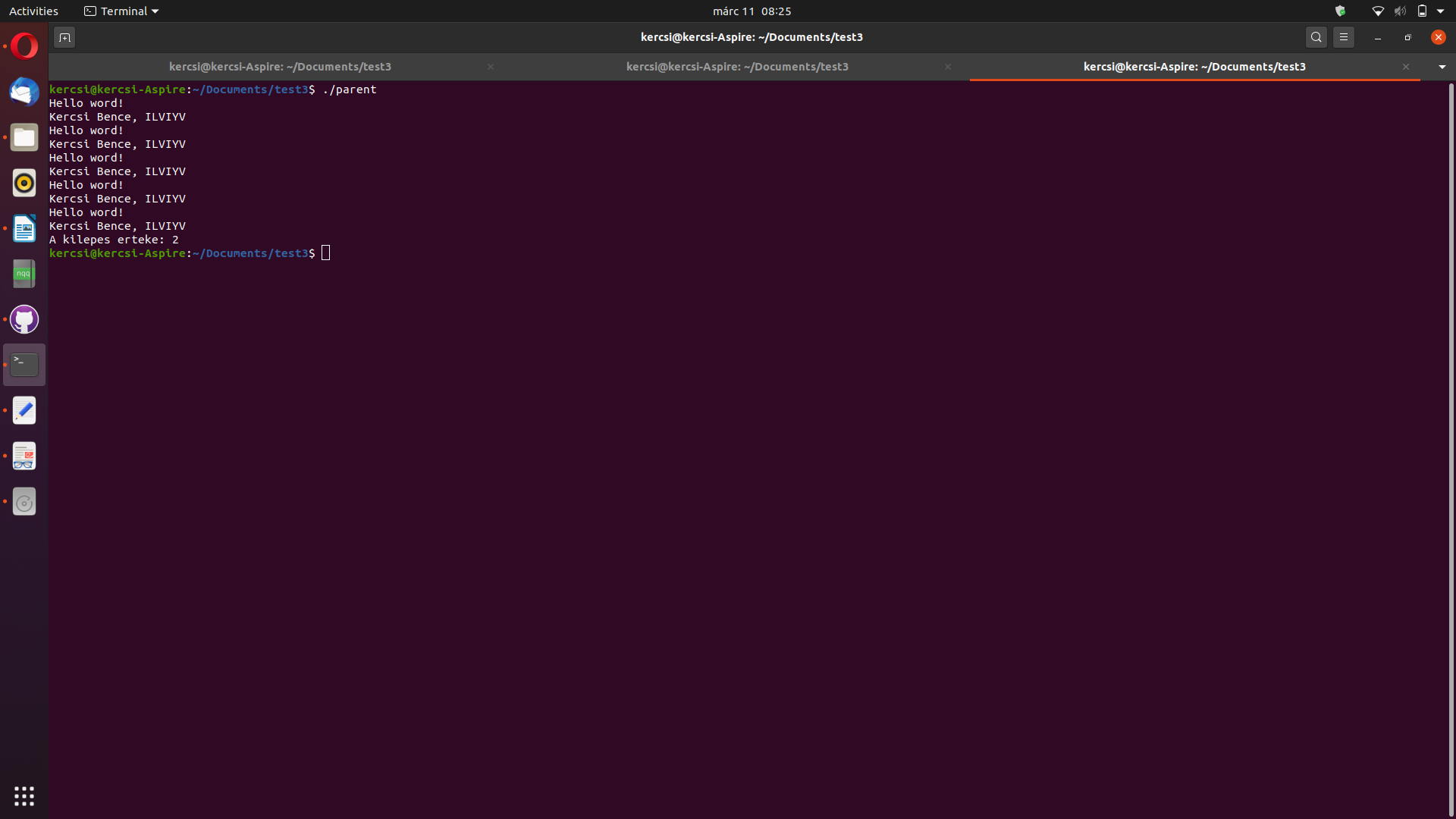
printf("Kercsi Bence, ILVIYV\n");

}

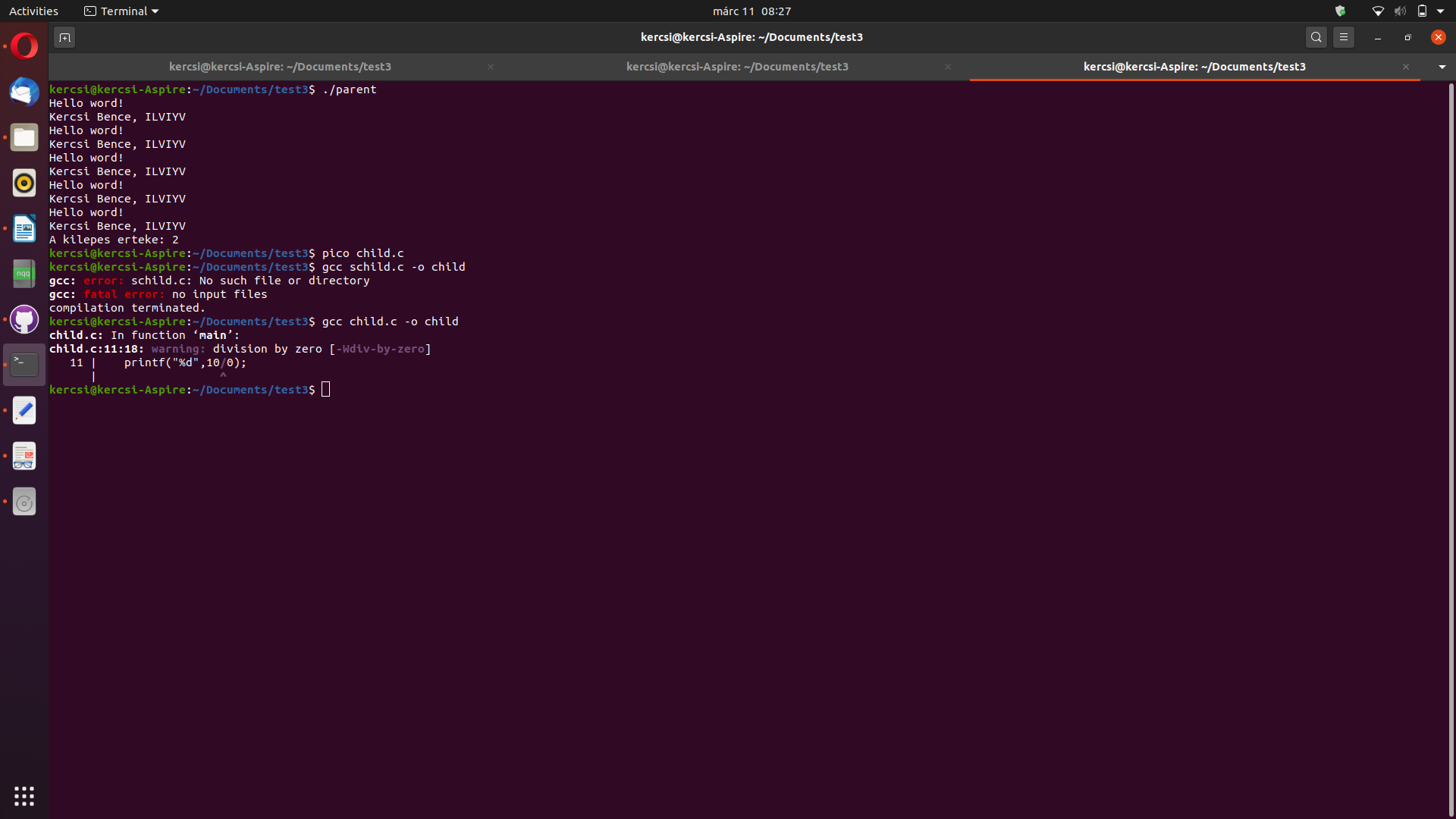
return 2;

}

Ez a 2 kód, de fel lesz töltve egyébként is. A child.c-ben azért van 2 megadva returnnek, hogy ellenőrizni tudjam, hogy jól veszi-e be a visszatérési értéket. Ezt a 2 feladatot egyben letudtam. Képek róla:

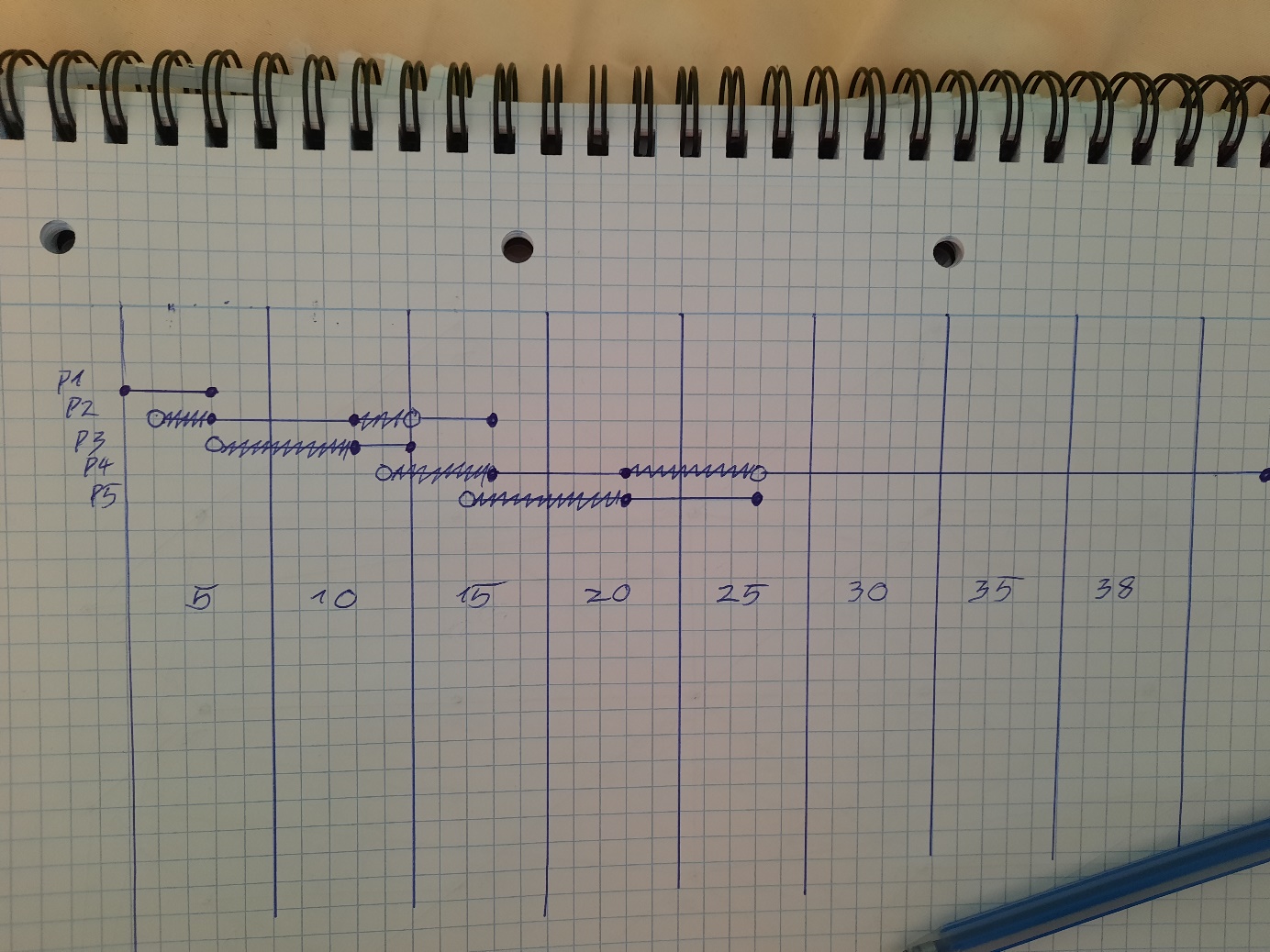
illetve módosítom úgy a child.c kódot, hogy 0-val osszon.

A 0-val való osztás olyan művelet, ami miatt nem lehet gcc-vel lefordítani a c programot.

6. a

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| RR:5ms | Round Robin | | | | |
| P1 | P2 | P3 | P4 | P5 |
| Érkezés | 0 | 1,8 | 3 | 9,18,28,33 | 12 |
| Cpu idő | 3 | 8,3 | 2 | 20,15,10,5 | 5 |
| Indulás | 0 | 3,10 | 8 | 13,23,28,33 | 18 |
| Befejezés | 3 | 8,13 | 10 | 18,28,33,38 | 23 |
| Várakozás | 0 | 2,2 | 5 | 4,5,0,0 | 6 |

P1→P2→P3→P2→P4→P5→P4\*



A recés rész az, ahol a processz várakozik ( nem tudtam, hogy hogy jelöljem).