

JEGYZŐKÖNYV

Operációs rendszerek BSc

2022. tavasz féléves feladat

Készítette: **Görög Krisztina Erzsébet**

Neptunkód: **MPW46D**

1.feladat – IPC mechanizmus

A feladat leírása:

A feladatom az volt, hogy írjak egy C programot, ami egy másodfokú egyenlet megoldóképletét reprezentálja nevesített csővezeték segítségével. A program a műveletvégzéshez szükséges adatokat egy bemeneti fájlból olvassa be, majd az adatokat és az eredményt visszaadja egy kimeneti fájlba. A két fájl struktúrája kötött.

A feladat elkészítésének lépései:

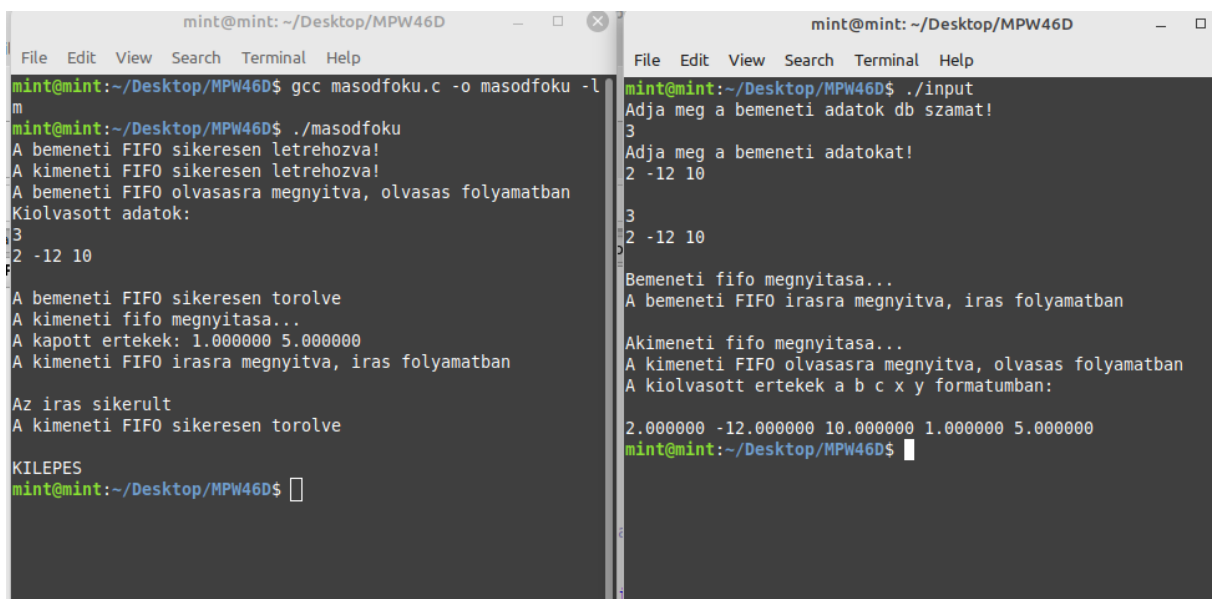
A feladatom szemléltetésére két programra volt szükségem; az egyik a masodfoku.c fájl, ami a bemeneti fájlból származó adatokkal kiszámítja az eredményt és fájlba írja azt, a másik az input.c fájl, ami bekéri tőlünk a másodfokú egyenlet paramétereit, majd ezeket a bemeneti fájlba írja, és miután a masodfoku.c kiszámolta az eredményeket, kiolvassa a kimeneti fájlból azokat.

Az input.c működése a következők szerint történik. Először bekéri a bemeneti adatok db számát (i) int-ként, majd egy stringként (buf) bekéri a bemeneti adatokat. Ezeket egy sztringbe (input) összefűzi, majd ezt fogja beleírni a bemeneti fájlba. Miután a masodfoku.c beírta az eredményeket a kimeneti fájlba, megnyitja és kiolvassa belőle az eredményt (ez a solution string).

A masodfoku.c működése a következő. Először is létrehozza a bemeneti és kimeneti fájlokat, majd kiolvassa a bemeneti fájl tartalmát (input string), törli a fájlt. Az input stringet szétszedjük: az első benne lévő szám lesz a bemeneti adatok db száma (i), a többi részével egyelőre nem foglalkozunk, ők a bemeneti adatok (bemeneti_adatok string). Az i-től függően fogjuk felbontani a bemeneti_adatok stringet értelemszerűen a, b, c változókra, tehát például, ha $i = 2$, akkor a bemeneti_adatok csak az a és b változókat tartalmazzák, a c-t nem kell kiolvasnunk, hanem nullára állítjuk. A nem megadott változókat nullára állítjuk. Ezután megnyitjuk a kimeneti fájlt írásra, létrehozunk egy solution nevű stringet a megoldás tárolására. Kiszámoljuk az egyenlet diszkriminánsát, ha kisebb, mint nulla, akkor a solutionba a bemeneti adatok mellé azt írjuk, hogy az egyenletnek nincs valós gyöke. Ha nagyobb, vagy egyenlő nullával, és $i = 1$, akkor mindkét megoldást nullára állítjuk, hisz ilyenkor fölösleges számolnunk, mindig nullát kapunk az a-tól függetlenül. Ha $i \neq 1$, akkor a megoldóképlet alapján kiszámítjuk a x-et és y-t. A solutionba összefűzzük a bemeneti adatokat és a két megoldást, ezt írjuk bele a kimeneti fájlba. Végül megszüntetjük a kimeneti fájlt.

A futtatás eredménye:

Három bemeneti adat esetén:



```
mint@mint: ~/Desktop/MPW46D
File Edit View Search Terminal Help
mint@mint:~/Desktop/MPW46D$ gcc masodfoku.c -o masodfoku -lm
mint@mint:~/Desktop/MPW46D$ ./masodfoku
A bemeneti FIFO sikeresen létrehozva!
A kimeneti FIFO sikeresen létrehozva!
A bemeneti FIFO olvasásra megnyitva, olvasas folyamatban
Kiolvasott adatok:
3
2 -12 10
A bemeneti FIFO sikeresen torolve
A kimeneti fifo megnyitasa...
A kapott ertekek: 1.000000 5.000000
A kimeneti FIFO irasra megnyitva, iras folyamatban
Az iras sikerult
A kimeneti FIFO sikeresen torolve
KILEPES
mint@mint:~/Desktop/MPW46D$

mint@mint: ~/Desktop/MPW46D
File Edit View Search Terminal Help
mint@mint:~/Desktop/MPW46D$ ./input
Adja meg a bemeneti adatok db szamat!
3
Adja meg a bemeneti adatokat!
2 -12 10
3
2 -12 10
Bemeneti fifo megnyitasa...
A bemeneti FIFO irasra megnyitva, iras folyamatban
A kimeneti fifo megnyitasa...
A kimeneti FIFO olvasasra megnyitva, olvasas folyamatban
A kiolvasott ertekek a b c x y formatumban:
2.000000 -12.000000 10.000000 1.000000 5.000000
mint@mint:~/Desktop/MPW46D$
```

Kettő bemeneti adat esetén:

```
mint@mint: ~/Desktop/MPW46D
File Edit View Search Terminal Help
mint@mint:~/Desktop/MPW46D$ ./masodfoku
A bemeneti FIFO sikeresen létrehozva!
A kimeneti FIFO sikeresen létrehozva!
A bemeneti FIFO olvasásra megnyitva, olvasas folyamatban
Kiolvasott adatok:
2
1 -3
A bemeneti FIFO sikeresen torolve
A kimeneti fifo megnyitasa...
A kapott ertekek: 0.000000 3.000000
A kimeneti FIFO irasra megnyitva, iras folyamatban

Az iras sikerult
A kimeneti FIFO sikeresen torolve

KILEPES
mint@mint:~/Desktop/MPW46D$
```

```
mint@mint: ~/Desktop/MPW46D
File Edit View Search Terminal Help
mint@mint:~/Desktop/MPW46D$ ./input
Adja meg a bemeneti adatok db szamat!
2
Adja meg a bemeneti adatokat!
1 -3
2
1 -3

Bemeneti fifo megnyitasa...
A bemeneti FIFO irasra megnyitva, iras folyamatban

Akimeneti fifo megnyitasa...
A kimeneti FIFO olvasásra megnyitva, olvasas folyamatban
A kiolvasott ertekek a b c x y formatumban:

1.000000 -3.000000 0.000000 0.000000 3.000000
mint@mint:~/Desktop/MPW46D$
```

Egy bemeneti adat esetén:

```
mint@mint: ~/Desktop/MPW46D
File Edit View Search Terminal Help
mint@mint:~/Desktop/MPW46D$ ./masodfoku
A bemeneti FIFO sikeresen létrehozva!
A kimeneti FIFO sikeresen létrehozva!
A bemeneti FIFO olvasásra megnyitva, olvasas folyamatban
Kiolvasott adatok:
1
5
A bemeneti FIFO sikeresen torolve
A kimeneti fifo megnyitasa...
A kapott ertekek: 0.000000 0.000000
A kimeneti FIFO irasra megnyitva, iras folyamatban

Az iras sikerult
A kimeneti FIFO sikeresen torolve

KILEPES
mint@mint:~/Desktop/MPW46D$
```

```
mint@mint: ~/Desktop/MPW46D
File Edit View Search Terminal Help
mint@mint:~/Desktop/MPW46D$ ./input
Adja meg a bemeneti adatok db szamat!
1
Adja meg a bemeneti adatokat!
5
1
5

Bemeneti fifo megnyitasa...
A bemeneti FIFO irasra megnyitva, iras folyamatban

Akimeneti fifo megnyitasa...
A kimeneti FIFO olvasásra megnyitva, olvasas folyamatban
A kiolvasott ertekek a b c x y formatumban:

5.000000 0.000000 0.000000 0.000000 0.000000
mint@mint:~/Desktop/MPW46D$
```

Ha az egyenletnek nincs megoldása:

```
mint@mint: ~/Desktop/MPW46D$ ./masodfoku
A bemeneti FIFO sikeresen létrehozva!
A kimeneti FIFO sikeresen létrehozva!
A bemeneti FIFO olvasásra megnyitva, olvasas folyamatban
Kiolvasott adatok:
3
3 4 5
A bemeneti FIFO sikeresen torolve
A kimeneti fifo megnyitasa...
Az egyenletnek nincs valos gyoke!
A kimeneti FIFO irasra megnyitva, iras folyamatban

Az iras sikerult
A kimeneti FIFO sikeresen torolve

KILEPES
mint@mint:~/Desktop/MPW46D$
```

```
mint@mint: ~/Desktop/MPW46D$ ./input
Adja meg a bemeneti adatok db szamat!
3
Adja meg a bemeneti adatokat!
3 4 5
3
3 4 5

Bemeneti fifo megnyitasa...
A bemeneti FIFO irasra megnyitva, iras folyamatban

Akimeneti fifo megnyitasa...
A kimeneti FIFO olvasásra megnyitva, olvasas folyamatban
A kiolvasott ertekek a b c x y formatumban:

3.000000 4.000000 5.000000 Az egyenletnek nincs valos gyoke!
mint@mint:~/Desktop/MPW46D$
```

2.feladat – OS algoritmusok

A feladat leírása:

A megadott terhelés mellett kellett kiszámolnom RR: 5 ms ütemezési algoritmus esetén az indulás, befejezés, várakozás, átlagos várakozás, körülfordulás, átlagos körülfordulás, válaszidő, átlagos válaszidő értékeket és a CPU kihasználtságot. (Tudjuk a következőket: cs: 0,1 ms; sch: 0,1 ms.) Az aktív/várakozó processzek menetét Gantt diagram segítségével kellett ábrázolnom.

A feladat elkészítésének lépései:

A Round Robin algoritmusnak megfelelően a listánkban legrégebb óta megérkezett vagy várakozó processznek adjuk meg a futási engedélyt, a váltás 5 ms-enként lehetséges. Váltásnál megnézzük, hogy van-e az előzőleg futott processzünkön kívül másik processz, ha igen, akkor azt futtatjuk le, ha több processz is van, akkor a legrégebb óta várakozót. Ha nincs más processzünk, akkor végig futtathatjuk.

A várakozási időt az érkezés és indulás között eltelt idő kiszámolásával kapjuk meg, ha ezen értékeket összeadjuk, és elosztjuk a processzek számával, akkor megkapjuk az átlagukat. A körülfordulási időt úgy kapjuk meg, hogy összeadjuk a processz várakozási és CPU idejét, az átlagos körülfordulási időt úgy, hogy az összes körülfordulási időt osztjuk a processzek számával. A válaszidő a processz érkezése és első futásának kezdete között eltelt idő, tehát az első várakozási idővel egyenlő. A válaszidők összegének az összes processz számával való osztásával megkapjuk a válaszidők átlagát. A CPU kihasználtságot úgy kapjuk meg, hogy az összes CPU időt osztjuk az összes CPU idő és a cs, valamint sch számának összegével. Tehát: $\text{összes CPU idő} / (\text{összes CPU idő} + (\text{befejezett processzek száma} * \text{sch}) + (\text{processzváltás} * \text{cs}))$. Cs-et annyiszor kell venni, ahányszor processzet váltunk (context switch), sch-t annyiszor, ahányszor befejeződik egy processz.

A feladat végeredménye:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	RR: 5 ms	P1	P2	P3	P4									
2	Érkezés	0, 10	8, 15	12, 25, 37	20, 32									
3	CPU idő	15	7	26	10									
4	Indulás	0, 15	10, 25	20, 32, 42	27, 37									
5	Befejezés	10, 20	15, 27	25, 37, 58	32, 42									
6	Várakozás	0, 5	2, 10	8, 7, 5	7, 5									
7	Körülfordulási idő	20	19	46	22									
8	Válaszidő	0	2	8	7									
9	Processzek végrehajtásának sorrendje: P1-P2-P1-P3-P2-P4-P3-P4-P3													
10														
11	CPU kihasználtság	$58 / (58 + (4 * 0,1 + 8 * 0,1)) = 58 / 59,2 = 97,97\%$												
12	Körülfordulási idők átlaga	$(20 + 19 + 46 + 22) / 4 = 107 / 4 = 26,75$												
13	Várakozási idők átlaga	$(5 + 12 + 20 + 12) / 4 = 49 / 4 = 12,25$												
14	Válaszidők átlaga	$(0 + 2 + 8 + 7) / 4 = 17 / 4 = 4,25$												
15														
16														

