

JEGYZŐKÖNYV

Operációs rendszerek BSc

2022. tavasz féléves feladat

Készítette: **Hajdu Adrián**

Neptunkód: **UY5E1L**

1. IPC

A feladat leírása:

18. Írjon egy olyan C programot, ami egy bemeneti fájlból 3 adatot olvas ki (háromszög oldalai) és eldönti, hogy szerkeszthető-e belőlük háromszög. A feladat megoldása során használjon message queue (üzenetsoros mechanizmust), valamint a kimenet kerüljön egy másik fájlba. Ha szerkeszthető belőlük háromszög, adjon vissza 1-et, különben pedig 0-t. A ki/bemeneti fájl struktúrája kötött!

Példa a bemeneti és kimeneti fájl struktúrájára:

Bemeneti fájl:

x y z

Kimeneti fájl (A q jelzi a visszatérési értéket, tehát hogy szerkeszthető-e háromszög):

x y z q

A feladat elkészítésének lépései:

- Először létrehozok egy üzenetsort (message queue), ehhez létrehozom a szükséges változókat, az üzenetsor azonosítót...
- Ezután beolvasom a háromszög oldalait a fájlból egy sztringbe
- A következő lépésben másolatot készítek a sztringről a sendbufferben lévő sztringbe, illetve ez alapján beállítom az üzenet méretét
- Mindezek után pedig beküldöm az msgsnd()-vel az üzenetet a queue-ba
- Ezután egy külön függvényben előkészítem a üzenetsort a fogadásra
- Megnyitom a fájlt, amelybe az üzenetet szeretném kiolvasni a queue-ból
- A bufferben lévő sztringbe kiolvastatom az üzenetet
- Létrehozok egy integer tömböt, amiben tárolom majd az üzenetben érkezett értékeket
- A tömbbe beolvasom az értékeket, közben sztringről integerre konvertálom
- Elkészítem az ellenőrzést arra, hogy megszerkeszthető-e a háromszög, majd kiíratom az eredményt egy új fájlba
- Végül kiürül az üzenetsor, s a program futása leáll

Kieg.:

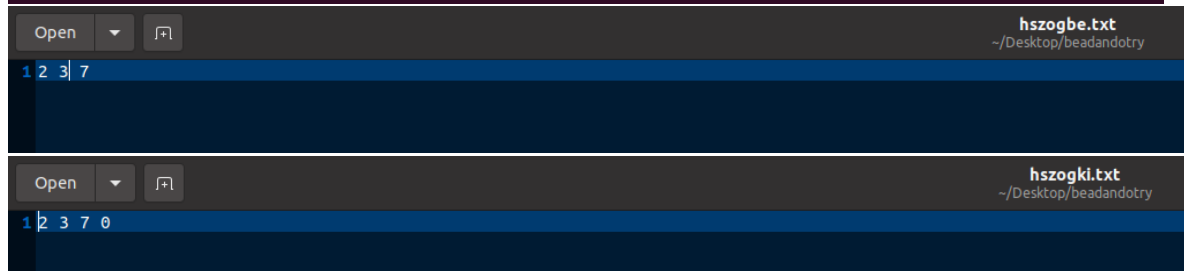
- Külön függvényt írtam a „haromszogbe.txt” feltöltéséhez (manuálisan is létrehozható, a program futását nem befolyásolja, csak ki kell kommentelni a függvényt)

A futtatás eredménye:

a, Ha olyan oldalakat adunk meg, amelyből háromszög nem szerkeszthető:

```
adrian@adrian-VirtualBox:~/Desktop/beadandotry$ gcc haromszog.c -o haromszog
adrian@adrian-VirtualBox:~/Desktop/beadandotry$ ./haromszog
Adja meg a haromszog 3 oldalat (pl. 3 5 7)
2 3 7
.
.
.
A program sikeresen lefutott!

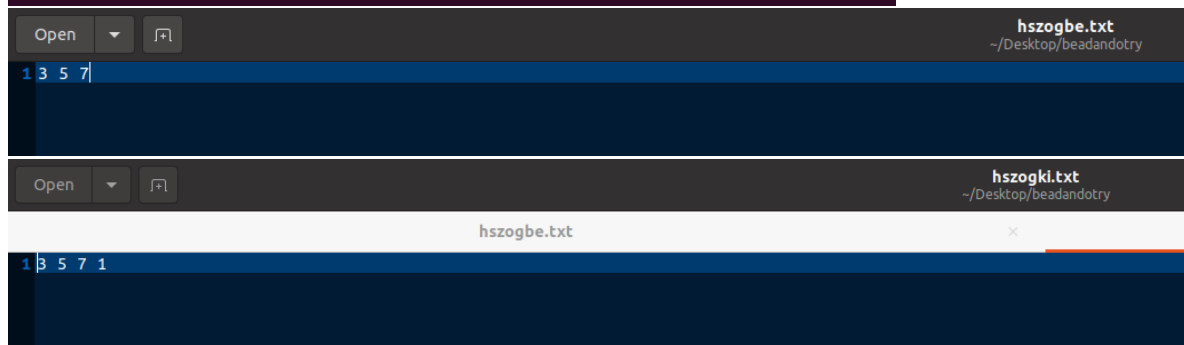
A haromszog: nem megszerkesztheto!
adrian@adrian-VirtualBox:~/Desktop/beadandotry$
```



b, Ha olyan oldalakat adunk meg, amelyből a háromszög megszerkeszthető:

```
adrian@adrian-VirtualBox:~/Desktop/beadandotry$ ./haromszog
Adja meg a haromszog 3 oldalat (pl. 3 5 7)
3 5 7
.
.
.
A program sikeresen lefutott!

A haromszog: megszerkesztheto!
adrian@adrian-VirtualBox:~/Desktop/beadandotry$
```



2. Ütemezés

A feladat leírása:

8. Adott egy rendszerbe az összes osztály-erőforrások száma: R (R1: 8; R2: 9; R3: 13)

A rendszerbe 4 processz van: P1, P2, P3, P4.

Teljesíthető-e P2 (1, 2, 2) kérése?

Biztonságos-e vagy nem biztonságos holtpontmentesség szempontjából a rendszer - a következő kiinduló állapot alapján?

- Határozza meg a processzek által igényelt erőforrások mátrixát – P2 processz kérésének figyelembe vételével?
- Határozza meg pillanatnyilag szabad erőforrások számát?
- Igazolja az egyes processzek végrehajtásának lehetséges sorrendjét - számolással?

Kiinduló állapot							
	Max. igény				Foglal		
	R1	R2	R3		R1	R2	R3
P1	4	2	5	P1	2	2	3
P2	7	7	7	P2	0	1	1
P3	1	4	3	P3	1	2	2
P4	3	7	4	P4	2	1	2

A feladat elkészítésének lépései:

- A bankár algoritmust alkalmazom
- A MAX. IGÉNY-ből kivonom a FOGLAL-t, így létrejön az IGÉNY mátrix
- Összeadom a FOGLAL oszlopaiban szereplő értékeket, majd ezeket kivonom az osztály-erőforrások számából, így megkapom a készletet
- Megnézem, hogy van-e olyan processz, amely igénye kielégíthető a készletből
- Ha megtaláltam melyik kielégíthető, abban az esetben a továbbiakban azzal már nem kell foglalkozni, mert ki lett elégítve
- A kielégített processzhez tartozó foglalat hozzáadom az eddigi készlethez, így létrejön egy új készlet
- Az előző 3 lépést ismétlem egészen addig, amíg minden processz igénye ki nem elégült
- Végül azt kapom, hogy ebben a felállásban minden processz igénye kielégíthető (Pl. P1 -> P3 -> P4 -> P2)

- Tehát P2 kérése kielégíthető, illetve a rendszer holtpontmentesség szempontjából biztonságos

Eredmény:

Az összes osztály- erőforrások száma: (8, 9, 13)												
Kiinduló állapot												
1. lépés				2. lépés			P2 kérése (1,2,2)					
MAX. IGÉNY				FOGLAL			IGÉNY					
	R1	R2	R3		R1	R2	R3		R1	R2	R3	
P1	4	2	5		2	2	3		2	0	2	P1 kielégíthető
P2	7	7	7		0	1	1		7	6	6	Új készlet: (5, 5, 8)
P3	1	4	3		1	2	2		0	2	1	
P4	3	7	4		2	1	2		1	6	2	
				5	6	8	Készlet: (3, 3, 5)					
MAX. IGÉNY				FOGLAL			IGÉNY					
	R1	R2	R3		R1	R2	R3		R1	R2	R3	
P1	4	2	5		2	2	3					P3 kielégíthető
P2	7	7	7		0	1	1		7	6	6	Új készlet: (6, 7, 10)
P3	1	4	3		1	2	2		0	2	1	
P4	3	7	4		2	1	2		1	6	2	
				Készlet: (5, 5, 8)								
MAX. IGÉNY				FOGLAL			IGÉNY					
	R1	R2	R3		R1	R2	R3		R1	R2	R3	
P1	4	2	5		2	2	3					P4 kielégíthető
P2	7	7	7		0	1	1		7	6	6	Új készlet: (8, 8, 12)
P3	1	4	3		1	2	2					
P4	3	7	4		2	1	2		1	6	2	
				Készlet: (6, 7, 10)								
MAX. IGÉNY				FOGLAL			IGÉNY					
	R1	R2	R3		R1	R2	R3		R1	R2	R3	
P1	4	2	5		2	2	3					P2 kielégíthető
P2	7	7	7		0	1	1		7	6	6	Biztonságosan lefuttatható!
P3	1	4	3		1	2	2					
P4	3	7	4		2	1	2					