

# Jegyzőkönyv

Operációs Rendszerek BSc

2021. tavasz féléves feladat

Készítette: **Mészáros Ákos**  
Neptunkód: **BFNA2**

## A feladat leírása:

Írjon egy olyan C programot, mely egy fájlból számpárokat kiolvassa meghatározza a legnagyobb közös osztóját.

A feladat megoldása során használjon shared memory(osztott memória szegmens) IPC mechanizmust, valamint a kimenet kerüljön egy másik fájlba. A kimeneti fájl struktúrája kötött!

Bemeneti fájl:

i (ez jelzi a számpárok darabszámát)

x y

...

Kimeneti fájl:

x y z (x,y a bemeneti adatokat jelzi, z pedig a kimenet eredményét)

## A feladat elkészítéseinek lépései:

```
int main(){
    int parent, child;
    FILE* fd;
    key_t key = ftok("shmfile",65);
    int shmid = shmget(key,32,0666|IPC_CREAT);

    parent = getpid();
    child = fork();
```

Első lépésként létrehozuk az osztott memória szegmenset, ezután fork()-al létrehozunk egy gyerek processzet.

```
if(child == 0){
    char buf[32];
    char *mem = (char*) shmatt(shmid,(void*)0,0);
    fd = fopen("bfna2x_in.txt", "r");
    if(!fd){
        perror("fopen(\"r\") hiba!");
        kill(parent, SIGKILL);
        exit(-1);
    }
    signal(SIGUSR1, handle_usrp);
```

```

if(child > 0){
    int n1, n2;
    char *mem = (char*) shmat(shmid, (void*)0, 0);
    char str[50];
    char *token;
    fd = fopen("bfna2x_out.txt", "w");
    if(!fd){
        perror("fopen(\"w\") hiba!");
        kill(child, SIGKILL);
        exit(-1);
    }
    signal(SIGINT, handle_int);
    signal(SIGUSR1, handle_usr);
}

```

Ezután a két processzert csatoljuk az osztott memória területéhez, a gyerek processzben megnyitjuk a bemeneti fájlt olvasásra, és a szülő processzel létrehozuk a kimeneti fájlt. Mindkét processzben beállítjuk a jelek kezelésével foglalkozó függvényeket.

```

signal(SIGUSR1, handle_usr);
fgets(buf, sizeof(buf), fd);

printf("%s", buf);

while(fgets(buf, sizeof(buf), fd)){
    printf("%s", buf);
    strcpy(mem, buf);
    kill(parent, SIGUSR1);
    while(!waitp){;}
    waitp = 0;
}

while(!stop){
    while(!waitc){;}
    waitc = 0;
    if(stop) break;
    strcpy(str, mem);
    kill(child, SIGUSR1);

    token = strtok(str, " ");
    n1 = atoi(token);
    token = strtok(NULL, " ");
    n2 = atoi(token);

    fprintf(fd, "%d %d ", n1, n2);
    while(n1!=n2){
        if(n1 > n2)
            n1 -= n2;
        else
            n2 -= n1;
    }
    fprintf(fd, "%d\n", n1);
}

```

A gyerek processz megkezd az olvasást, majd kiírja a beolvasott értékeket az osztott memória területre, miközben a szülő processz várakozik.

A gyerek az memóriába írás után jelet küld a szülőnek, aki az olvasás után

vissza jelez hogy a gyerek folytathatja az olvasást.

A szülő kiszámolja a legnagyobb közös osztót és kiírja fájlba az értékeket, amíg a gyerek már a következő értékeket olvassa.

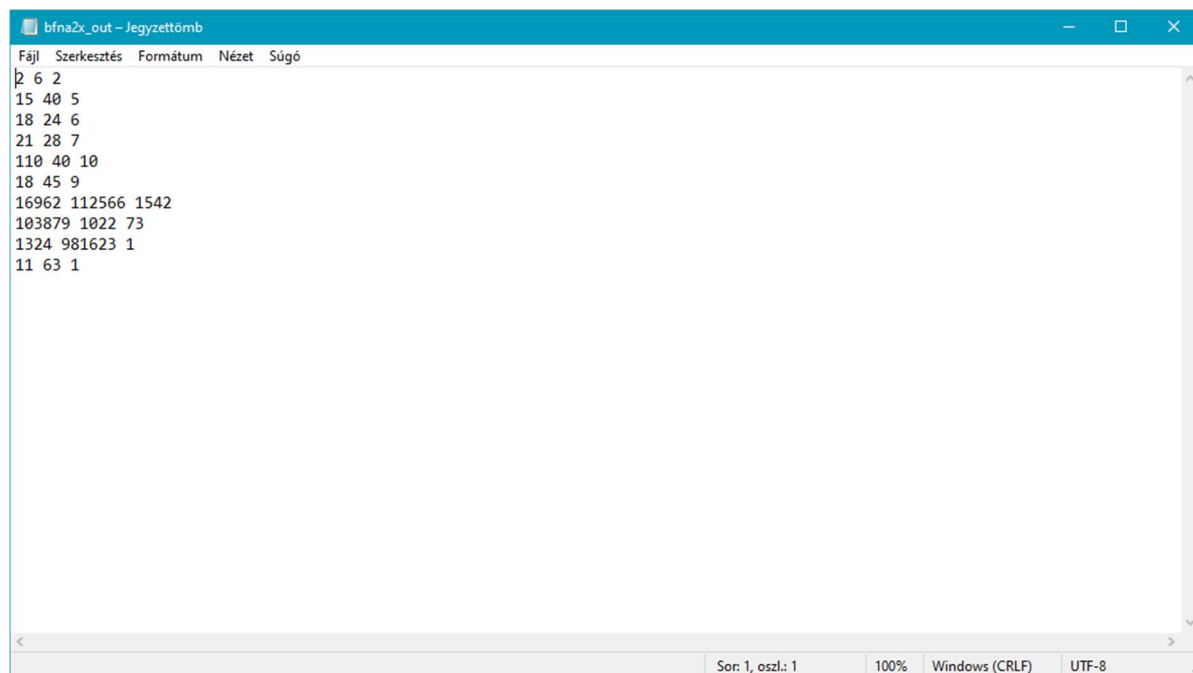
Miután mindkét processz befejezi a feladatait, újra várnak egymásra mielőtt újra kezdenék a loop-ot.

```
        shmdt(mem);  
        fclose(fd);  
        kill(parent, SIGINT);  
    }  
    shmdt(mem);  
    fclose(fd);  
    shmctl(shmid, IPC_RMID, NULL);  
}
```

Amikor a gyerek eléri a fájl végét, interrupt signalt küld a szülőnek, aki ettől tudni fogja hogy ne várjon újabb értékekre.

Mindkét processz lecsatlakozik az osztott memóriaterületről, bezárják a fájlfolyamokat és végül a szülő felszabadítja az osztott memóriaszegmensét.

## A futtatás eredménye:



```
bfna2x_out - Jegyzettömb  
2 6 2  
15 40 5  
18 24 6  
21 28 7  
110 40 10  
18 45 9  
16962 112566 1542  
103879 1022 73  
1324 981623 1  
11 63 1  
Sor: 1, oszl.: 1 100% Windows (CRLF) UTF-8
```

```
blackhun@BlackHUN-VBMin: ~/Desktop/osgyak/BFNA2XOsGyak/OSSemTask_BFNA...  
File Edit View Search Terminal Help  
blackhun@BlackHUN-VBMin:~/Desktop/osgyak/BFNA2XOsGyak/OSSemTask_BFNA2X$ ./bfna2  
x_17  
10  
2 6  
write finished 1  
read finished 1  
15 40  
write finished 1  
read finished 1  
18 24  
write finished 1  
read finished 1  
21 28  
write finished 1  
read finished 1  
110 40  
write finished 1  
read finished 1  
18 45  
write finished 1  
read finished 1  
16962 112566  
write finished 1  
read finished 1  
103879 1022  
write finished 1  
read finished 1  
1324 981623  
write finished 1  
read finished 1  
11 63  
write finished 1  
read finished 1
```