



10. FELADAT

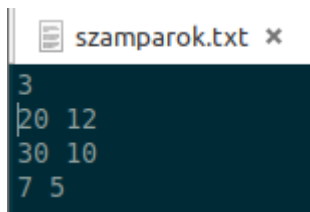
Készítette: Ferencsik Márk (UJTWLL)



A feladat leírása: Egy előre meghatározott tartalmú fájlból olvassa be a számpárokat, majd számolja ki a legnagyobb közös osztójukat, az eredményt pedig írja ki egy fájlba a számpárokkal együtt. A feladat megoldása során használjon IPC-s üzenetküldő mechanizmust.

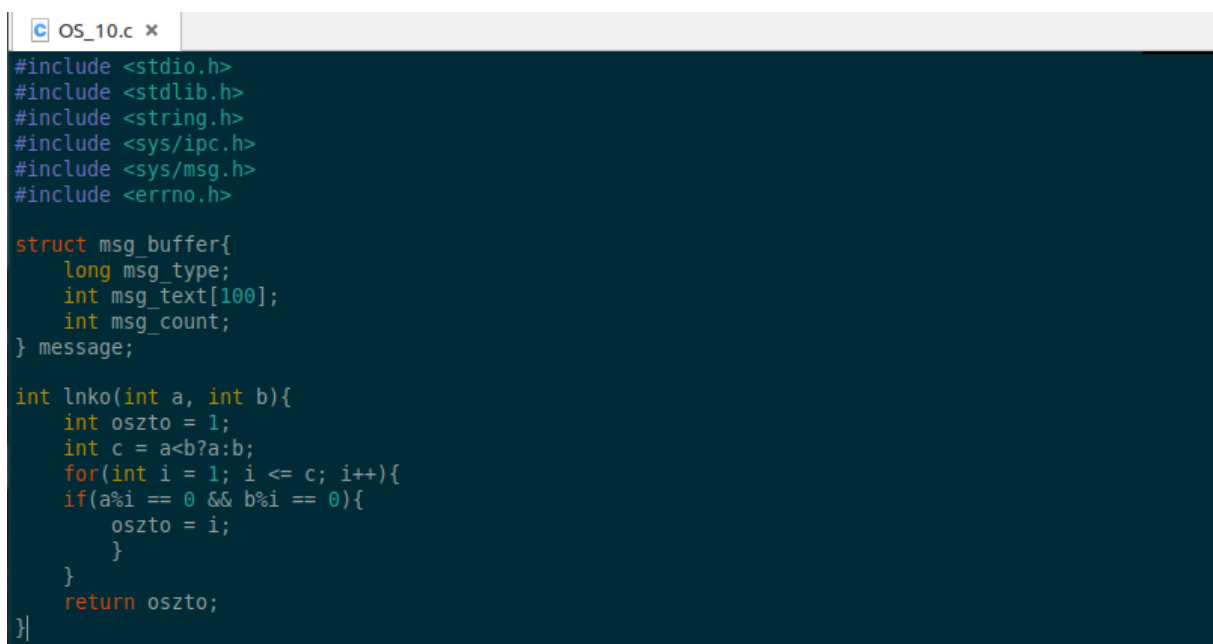
A program leírása:

Alapvető feltétel, hogy létezzen egy szöveges állomány, ami tárolja a számpárok számát, valamint magukat a számpárokat. Ezért létrehoztam egy „szamparok.txt” állományt.



```
szamparok.txt x
3
20 12
30 10
7 5
```

Maga a forráskód:



```
OS_10.c x
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <errno.h>

struct msg_buffer{
    long msg_type;
    int msg_text[100];
    int msg_count;
} message;

int loko(int a, int b){
    int osztó = 1;
    int c = a<b?a:b;
    for(int i = 1; i <= c; i++){
        if(a%i == 0 && b%i == 0){
            osztó = i;
        }
    }
    return osztó;
}
```

IPC-s üzenetküldő mechanizmusnak struktúrát hoztam létre, valamint létrehoztam egy függvényt, ami kiszámolja két integerként kapott szám legnagyobb közös osztóját. Ezután következik a main() függvény.

```

OS_10.c x
int main(){
    key_t key;
    int msgid;
    int db = 0;
    int szamparCount;
    int msg_db = 0;
    char *sor;

    FILE *fadat = fopen("szamparok.txt", "r");

    if(fadat == NULL){
        fprintf(stderr, "Sikertelen fajlmegnyitas (%s)\n", strerror(errno));
    }
    fscanf(fadat, "%d", &szamparCount);

    int a[szamparCount];
    int b[szamparCount];
    int c[szamparCount];

    for(int i = 0; i <= 2; i++){
        fscanf(fadat, "%d %d", &a[i], &b[i]);
        printf("%d %d\n", a[i], b[i]);
        db++;
    }
}

```

Itt először előkészítem a változókat, valamint beolvasom az adott fájlból először a számpárok számát, utána a számpárokat. Amennyiben nem sikerült megnyitnia a fájlt, akkor értesítést kapunk a hibakimeneten. Ugyanakkor sikeres fájlmegnyitás esetén megtörténik az adatok beolvasása egy for ciklusban.

```

OS_10.c x
key = ftok("msg.txt", 100);
msgid = msgget(key, 0666 | IPC_CREAT);
message.msg_type = 1;

for(int j = 0; j < szamparCount; j++){
    c[j] = lnko(a[j], b[j]);
    printf("%d es %d kozos osztója: %d\n", a[j], b[j], c[j]);
    message.msg_text[msg_db] = a[j];
    msg_db++;
    message.msg_text[msg_db] = b[j];
    msg_db++;
    message.msg_text[msg_db] = c[j];
    msg_db++;
    message.msg_count = msg_db;
}

msgsnd(msgid, &message, sizeof(message), 0);

FILE *fkiir = fopen("lnko.txt", "w+");
for(int k = 0; k < db; k++){
    fprintf(fkiir, "%d %d %d\n", a[k], b[k], c[k]);
}

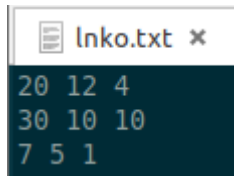
fclose(fkiir);

```

Ezután létrehozok egy kulcsot, amit az üzenetsor használni tud, valamint megszerzem az üzenetküldés ID-jét. Egy ciklusban minden számpárra kiszámoljuk a legnagyobb közös nevezőt, majd az eredmény eltároljuk az üzenetküldés struktúrájában. Megtörténik az üzenet elküldése. Utolsó

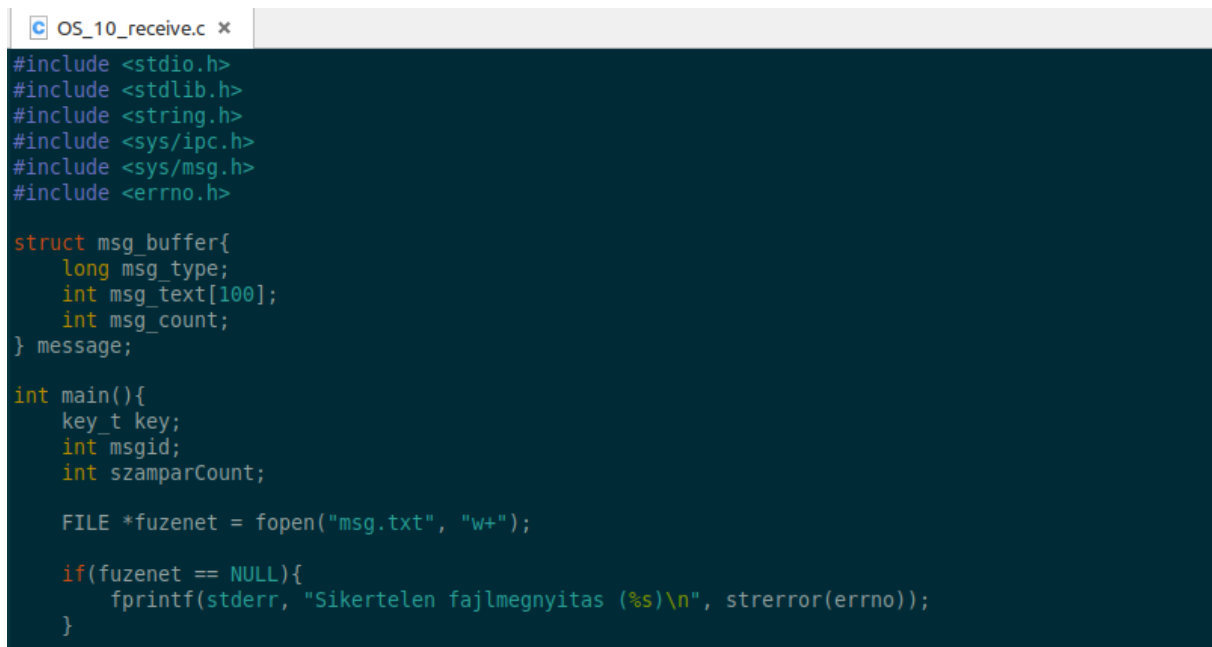
lépésben létrehozunk egy „lnko.txt” nevű kimeneti fájlt, amibe az eredményeket kiírhatjuk, majd megtörténik a fájlba írás. Ezzel a program befejezi a futását.

A kimeneti fájl felépítése:



```
lnko.txt x
20 12 4
30 10 10
7 5 1
```

Létrehoztam egy „OS_10_receive.c” nevű programot is, mellyel felfoghatjuk az „OS_10.c”-ben küldött üzenetet.



```
OS_10_receive.c x
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <errno.h>

struct msg_buffer{
    long msg_type;
    int msg_text[100];
    int msg_count;
} message;

int main(){
    key_t key;
    int msgid;
    int szamparCount;

    FILE *fuzenet = fopen("msg.txt", "w+");

    if(fuzenet == NULL){
        fprintf(stderr, "Sikertelen fajlmegnyitas (%s)\n", strerror(errno));
    }
}
```

Itt is létrehozzuk az IPC-s üzenetküldő mechanizmusnak a struktúrát, mivel ebben fogjuk ideiglenesen letárolni, majd egy „msg.txt” fájlba kiírni az üzenetet. Majd jön a main() függvény.

Itt szintén létrehozzuk a szükséges változókat, valamint megnyitjuk az „msg.txt” állományt. Hibás fájlmegnyitás esetén itt is kapunk értesítést.

```

key = ftok("msg.txt", 100);
msgid = msgget(key, 0666 | IPC_CREAT);
message.msg_type = 1;

msgrcv(msgid, &message, sizeof(message), 1, 0);
msgctl(msgid, IPC_RMID, NULL);

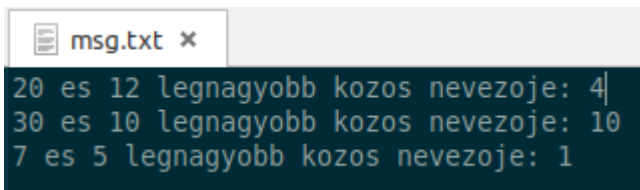
for(int i = 0; i < message.msg_count; i+=3){
    fprintf(fuzenet, "%d es %d legnagyobb kozos nevezoje: %d\n", message.msg_text[i],
message.msg_text[i+1], message.msg_text[i+2]);
}

fclose(fuzenet);
return 0;
}

```

Legeneráljuk az üzenetküldés kulcsát, mely pontosan megegyezik az „OS_10.c” nevű programban létrehozott kulccsal. Megkapjuk az üzenetküldés azonosítóját, ezután fogadjuk, majd töröljük az üzenetet a sorból. Egy for ciklusban pedig kiíratjuk a formázott üzenetet a kimeneti „msg.txt” fájlba, majd bezárjuk a fájlt, és ez a program is befejezi a futását.

A „msg.txt” kinézete:



```

msg.txt x
20 es 12 legnagyobb kozos nevezoje: 4
30 es 10 legnagyobb kozos nevezoje: 10
7 es 5 legnagyobb kozos nevezoje: 1

```