Լաբորատոր աշխատանք 1

Inter-process Communication (IPC)
Հաղորդագրությունների հերթեր (Message Queues)

Հաղորդագրությունների հերթը պրոցեսների հաղորդակցության մեխանիզմ է, որը պրոցեսներին հնարավորություն է տալիս կատարել տվյալների փոխանակում: Տվյալների փոխանակումն իրականացվում է ամբողջական հաղորդագրությունների տեսքով. հնարավոր չէ հերթից իրականացնել հաղորդագրության մասնակի ընթերցում՝ մնացած մասը հերթում թողնելով, կամ իրականացնել մի քանի հաղորդագրությունների միաժամանակ ընթերցում: Հաղորդագրությունները հերթում տեղադրվում և ընթերցվում են FIFO (First in, first out) սկզբունքով:

Հերթի ստեղծումը

msgget() համակարգային կանչը ստեղծում է նոր հաղորդագրությունների հերթ կամ ստանում է արդեն ստեղծված հերթի id-ն: Ֆունկցիայի պրոտոտիպը հետևյալն է.

```
#include <sys/msg.h>
int msgget(key_t key, int msgflag);
```

- **key** – բանալի, որը կարելի է գեներացնել **IPC_PRIVATE** հաստատունի կամ **ftok()** ֆունկցիայի օգնությամբ
- **msgflag** – թույլտվության բիթեր (permission bits)

Constant	Octal value	Permission bit
S_IRUSR	0400	User-read
S_IWUSR	0200	User-write
S_IXUSR	0100	User-execute
S_IRGRP	040	Group-read
S_IWGRP	020	Group-write
S_IXGRP	010	Group-execute
S_IROTH	04	Other-read
S_IWOTH	02	Other-write
S_IXOTH	01	Other-execute

Բացի վերը նշված արժեքներից, msgflag-ին կարող են bitwise OR գործողությամբ ավելանալ հետևյալ դրոշակները, որոնք ղեկավարում են msgget ֆունկցիայի կողմից կատարվող գործողությունը.

- **IPC_CREAT** – եթե չկա տրված key-ով գոյություն ունեցող հերթ, ապա ստեղծել նորը,
- **IPC_EXCL** – եթե նշված է IPC_CREAT դրոշակը, և տրված բանալիով հերթ արդեն գոյություն ունի, ապա կանչն ավարտվում է EEXIST error-ով:

message_create.c ծրագիրը msgget() կանչի միջոցով ստեղծում է նոր հաղորդագրությունների հերթ: Ծրագիրը կատարելու համար անհրաժեշտ է.

```
gcc message_create.c init_queue.c -o message_create
./message_create
```

Ստեղծված հերթերը կարելի է ստուգել **ipcs –q** հրամանի միջոցով:

Հաղորդագրությունների տեղադրումը հերթում

Հերթում գրելու համար կիրառվում է **msgsnd()** համակարգային կանչը: Ֆունկցիայի պրոտոտիպը հետևյալն է.

```
#include <sys/msg.h>
int msgsnd(int msqid, const void *msgp, size_t msgsz, int msgflg);
```

- **msqid** – Հերթի id,
- **msgp** – Ծրագրավորողի կողմից հայտարարված ստրուկտուրայի օբյեկտ (օրինակը ներկայցված է message_queue.h ֆայլում),
- **msgsz** – Գրվող հաղորդագրության չափը՝ բայթերով,
- **msgflag** – Ղեկավարող bitmask: Կարող է լինել 0, կամ ընդունել IPC_NOWAIT արժեքը, որն իրականացնում է չարգելափակող կանչ:

msgsnd() կանչի օրինակը ներկայացված է message_send.c ծրագրում: Ծրագիրը կատարելու համար անհրաժեշտ է.

```
gcc message_send.c init_queue.c -o message_send
./message_send
```

Հաղորդագրությունների ընթերցումը հերթից

Հերթից կարդալու համար կիրառվում է **msgrcv()** համակարգային կանչը: Ֆունկցիայի պրոտոտիպը հետևյալն է.

```
#include <sys/msg.h>
ssize_t msgrcv(int msqid, void *msgp, size_t maxmsgsz, long msgtyp, int msgflg);
```

- **msqid** – Հերթի id,
- **msgp** – Ծրագրավորողի կողմից հայտարարված ստրուկտուրայի օբյեկտ,
- **maxmsgsz** – Կարդացվող հաղորդագրության առավելագույն չափը՝ բայթերով,
- **msgtyp** – Ստրուկտուրայում սահմանվաց հաղորդագրության տիպ,
- **msgflag** – Ղեկավարող bitmask: Կարող է լինել 0, կամ ընդունել հետևյալ 3 արժեքներից մեկը.
  - IPC_NOWAIT – Իրականացնել չարգելափակող կանչ:

- MSG_EXCEPT – Կարդալ առաջին հաղորդագրությունը, որի տիպը հավասար չէ msgtyp-ին:
- MSG_NOERROR – Եթե հաղորդագրության չափը մեծ է maxmsgsz-ից, ապա սխալ վերադարձնելու փոխարեն msgrcv ֆունկցիան վերադարձնում է հաղորդագրության առաջին maxmsgsz բայթերը:

msgrcv() կանչի օրինակը ներկայացված է message_receive.c ծրագրում: Ծրագիրը կատարելու համար անհրաժեշտ է.

```
gcc message_receive.c init_queue.c -o message_receive
./message_receive
```

Հաղորդագրությունների հերթի ղեկավարումը

Հերթի ղեկավարման համար կիրառվում է **msgctl()** ֆունկցիան: Ֆունկցիայի պրոտոտիպը հետևյալն է.

```
int msgctl(int msqid, int cmd, struct msqid_ds *buf);
```

cmd արգումենտը նշում է այն գործողությունը, որն իրականացվելու է: Ունի հնարավոր հետևյալ 3 արժեքները.

- IPC_RMID – անմիջապես ջնջել հաղորդագրությունների հերթը և դրա հետ կապված msqid_ds տվյալների ստրուկտուրան: Հերթում առկա բոլոր հաղորդագրությունները ջնջվում են, գրող (կարդացող) բոլոր պրոցեսները վերադարձնում են EIDRM error: 3-րդ արգումենտն անտեսվում է (Օրինակը ներկայացված է message_rm.c ծրագրում):
- IPC_STAT – պատճենել հաղորդագրությունների հերթի հետ կապված msqid_ds տվյալների ստրուկտուրան buf բուֆերի մեջ (Օրինակը ներկայացված է message_chqbyes.c ծրագրում):
- IPC_SET – Փոփոխել հերթի հետ կապված տվյալների msqid_ds ստրուկտուրայի նշված դաշտերը:

Տվյալների կառուցվածքը

Յուրաքանչյուր հաղորդագրությունների հերթ ունի իր հետ կապված տվյալների ստրուկտուրա.

```
struct msqid_ds {
   struct ipc_perm msg_perm; /* Ownership and permissions */
   time_t msg_stime; /* Time of last msgsnd() */
   time_t msg_rtime; /* Time of last msgrcv() */
   time_t msg_ctime; /* Time of last change */
```

```
    unsigned long __msg_cbytes; /* Number of bytes in queue */
    msgqnum_t msg_qnum; /* Number of messages in queue */
    msglen_t msg_qbytes; /* Maximum bytes in queue */
    pid_t msg_lspid; /* PID of last msgsnd() */
    pid_t msg_lrpid; /* PID of last msgrcv() */
};
```

Առաջադրանքներ

1. Կատարել message_create ծրագիրը, բացատրել աշխատանքի արդյունքում ցուցադրված հաղորդագրությունը:
2. Ստեղծված հերթի մեջ գրել նոր հաղորդագրություն, որը կունենա հետևյալ դաշտերը.
    - mtype = 20
    - mtext = "test"
3. **ipcs –q** հրամանով ստուգել հերթում գտնվող հաղորդագրությունների քանակը: Բացատրել **used-bytes** դաշտի արժեքը:
4. Հերթը դատարկել՝ կարդալով առկա բոլոր հաղորդագրությունները: Այնուհետև, ևս մեկ անգամ հերթից հաղորդագրություն կարդալու փորձ կատարել: Բացատրել ծրագրի աշխատանքի արդյունքը:
5. 4-րդ կետում կատարված ծրագրի աշխատանքը չընդհատելով, բացել նոր հրամանային տող և հերթում նոր հաղորդագրություն տեղադրել: Բացատրել նախորդ ծրագրի հրամանային տողում ցուցադրված հաղորդագրությունը:
6. Փոփոխել message_receive ծրագիրն այնպես, որ դատարկ հերթից կարդալու փորձի դեպքում ծրագրի աշխատանքը չարգելափակվի:
7. Փոփոխել message_receive ծրագիրն այնպես, որ սահմանված maxmsgsz չափից մեծ հաղորդագրություն կարդալու փորձի դեպքում սխալ տեղի չունենա:
8. Փոփոխել message_create ծրագիրն այնպես, որ յուրաքանչյուր հաջորդական աշխատանքի դեպքում ստեղծվի նոր հաղորդագրությունների հերթ:
9. message_rm ծրագրի միջոցով ջնջել առկա հերթերից մի քանիսը:
10. Հաղորդագրությունների հերթի հետ կապված msqid_ds ստրուկտուրայի msg_qbytes դաշտի արժեքը սահմանել 5 բայթ՝ օգտագործելով msg_chqbytes ծրագիրը: Այնուհետև հերթի մեջ ավելացնել նոր հաղորդագրություն հետևյալ պարունակությամբ՝ "Test message": Բացատրել ծրագրի աշխատանքի արդյունքը: