Operating Systems Lab
IPC-Message Passing

By: Muhammad Ahsan

1. Message Passing

Communication takes place by exchange of messages

If P & Q wish to communicate, they need to:

- Establish communication link between them
- Communication link can be uni/bi directional, and associated with a single pair of communicating processes
- Exchange messages via send(message), receive(message)

OS Message Queue is a linked list of messages. Queue identified by message queue identifier.

```
struct msg {
        long mtype;
        char mtext[MSGLENGTH];
};
```

This struct must be included in each process sharing messages.

Type = 0 receives next msg

Type = +ive receives next msg where type matches

Type = -ive receives 1 st msg where type < abs(-ive)

Example 1:

Process 1 (Sending Message)

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdio.h>
#include <string.h>
#define MSGSZ 128
typedef struct msgbuf {
        long mtype;
        char mtext[MSGSZ];
} message_buf;
int main()
{
	int msqid;
	int msgflg = IPC_CREAT | 0666;
	key_t key;
	message_buf sbuf;
	size_t buf_length;
	key = 1234;
	msqid = msgget(key, msgflg );
	sbuf.mtype = 1;
	strcpy(sbuf.mtext, "Did you get this?");
	buf_length = strlen(sbuf.mtext) + 1 ;
	msgsnd(msqid, &sbuf, buf_length, IPC_NOWAIT);
	return 0;
}
```

Process 2 (Receiving Message)

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdio.h>
#define MSGSZ 128
typedef struct msgbuf {
	long mtype;
	char mtext[MSGSZ];
} message_buf;
int main()
{
	int msqid;
	key_t key;
	message_buf rbuf;
	key = 1234;
	msqid = msgget(key, 0666);
	msgrcv(msqid, &rbuf, MSGSZ, 1, 0);
	printf("%s\n", rbuf.mtext);
	return 0;
}
```