SENDER: -

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/msg.h>
#define MAX_TEXT 512
struct my_msg_st {
long int my_msg_type;
char some_text[MAX_TEXT];};
int main(){
int running = 1;
struct my_msg_st some_data_bin;
struct my_msg_st some_data_oct;
struct my_msg_st some_data_hex;
int msgid;
char buffer[BUFSIZ];
msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
if (msgid == -1) {
fprintf(stderr, "msgget failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
while(running) {
printf("Enter The Decimal Number : ");
fgets(buffer, BUFSIZ, stdin);
some_data_bin.my_msg_type = 2;
some_data_oct.my_msg_type = 8;
some_data_hex.my_msg_type = 16;
if (strncmp(buffer, "end", 3) != 0)
//Decimal to Binary
```

```
int num = atoi(buffer);

long long int bin=0;

int i=1;

while (num!=0) {

int rem = num % 2;

num /= 2;

bin += rem * i;

i*= 10;

}

sprintf(some_data_bin.some_text,"%lld",bin);

// Decimal to octal

num = atoi(buffer);

long long int octal=0;

i=1;

while (num!=0) {

int rem = num % 8;

num /= 8;

octal += rem * i;

i*= 10;

}

sprintf(some_data_oct.some_text,"%lld",octal);

//Decimal to Hexadecimal

int decimalnum, quotient, remainder;

char hexadecimalnum[100]="";

quotient = atoi(buffer);

int j=0;

while (quotient != 0)

{

remainder = quotient % 16;

if (remainder < 10)

hexadecimalnum[j++] = 48 + remainder;
else
hexadecimalnum[j++] = 55 + remainder;
quotient = quotient / 16;
}
strcpy(some_data_hex.some_text,hexadecimalnum);
}
else{
strcpy(some_data_hex.some_text,buffer);
strcpy(some_data_bin.some_text,buffer);
strcpy(some_data_oct.some_text,buffer);
}
if (msgsnd(msgid, (void *)&some_data_bin, MAX_TEXT, 0) ==-1) {
fprintf(stderr, "msgsnd failed\n");
exit(EXIT_FAILURE);
}
if (msgsnd(msgid, (void *)&some_data_oct, MAX_TEXT, 0) ==-1) {
fprintf(stderr, "msgsnd failed\n");
exit(EXIT_FAILURE);
}
if (msgsnd(msgid, (void *)&some_data_hex, MAX_TEXT, 0) ==-1) {
fprintf(stderr, "msgsnd failed\n");
exit(EXIT_FAILURE);
}
if (strncmp(buffer, "end", 3) == 0) {
running = 0;
}
}
exit(EXIT_SUCCESS);
}
```

BINARY RECEIVER: -

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/msg.h>
struct my_msg_st {
long int my_msg_type;
char some_text[BUFSIZ];
};
int main(){
int running = 1;
int msgid;
struct my_msg_st some_data;
long int msg_to_receive = 2;
msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
if (msgid == -1) {
fprintf(stderr, "msgget failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
while(running) {
if (msgrcv(msgid, (void *)&some_data, BUFSIZ,
msg_to_receive, 0) == -1) {
fprintf(stderr, "msgrcv failed with error: %d\n",
errno);
exit(EXIT_FAILURE);
}
if (strncmp(some_data.some_text, "end", 3) == 0) {
running = 0;
printf("Program Terminated\n");
break;
}
printf("Decimal to Binary : %s\n",some_data.some_text);
}
exit(EXIT_SUCCESS);
}
```

OCTAL RECEIVER: -

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/msg.h>
struct my_msg_st {
long int my_msg_type;
char some_text[BUFSIZ];
};
int main(){
int running = 1;
int msgid;
struct my_msg_st some_data;
long int msg_to_receive = 8;
msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
if (msgid == -1) {
fprintf(stderr, "msgget failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
while(running) {
if (msgrcv(msgid, (void *)&some_data, BUFSIZ,
msg_to_receive, 0) == -1) {
```

```
fprintf(stderr, "msgrcv failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
if (strncmp(some_data.some_text, "end", 3) == 0) {
running = 0;
printf("Program Terminated\n");
break;
}
printf("Decimal to Octal : %s\n",some_data.some_text);
}
exit(EXIT_SUCCESS);
}
```

HEXADECIMAL RECEIVER: -

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/msg.h>
struct my_msg_st {
long int my_msg_type;
char some_text[BUFSIZ];
};
int main(){
int running = 1;
int msgid;
struct my_msg_st some_data;
long int msg_to_receive = 16;
msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
if (msgid == -1) {
```

```
fprintf(stderr, "msgget failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
while(running) {
if (msgrcv(msgid, (void *)&some_data, BUFSIZ,
msg_to_receive, 0) == -1) {
fprintf(stderr, "msgrcv failed with error: %d\n",
errno);
exit(EXIT_FAILURE);
}
if (strncmp(some_data.some_text, "end", 3) == 0) {
running = 0;
printf("Program Terminated\n");
break;
}
printf("Decimal to hexa : %s\n",some_data.some_text);
}
exit(EXIT_SUCCESS);
}
```