**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);

bin=0;

// Decimal to Octal

num = atoi(buffer);

long long int octal = 0;

int k = 1;

while (num != 0)

{

int rem = num % 8;

num /= 8;

octal += rem \* k;

k \*= 10;

}

sprintf(some\_data\_oct.some\_text, "%lld", octal);

octal=0;

// Decimal to HexaDecimal

int remainder;

int quotient = atoi(buffer);

char hexadecimalnum[1000];

char str[100];

int j = 0,l=0;

while (quotient != 0)

{

remainder = quotient % 16;

if (remainder < 10)

{

hexadecimalnum[j++] = 48 + remainder;

}

else

{

hexadecimalnum[j++] = 55 + remainder;

}

quotient = quotient / 16;

}

int len= strlen(hexadecimalnum);

l=len;

for(k=0;k<len;k++)

{

str[l-1]=hexadecimalnum[k];

l--;

}

sprintf(some\_data\_hex.some\_text,"%s",str);

}

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>

#define MAX\_TEXT 512

struct my\_msg\_st

{

long int my\_msg\_type;

char some\_text[MAX\_TEXT];

};

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);

}

printf("Decimal to Binary: %s\n",some\_data.some\_text);

if (strncmp(some\_data.some\_text, "end", 3) == 0)

{

running = 0;

printf("Program Terminated\n");

break;

}

}

if(msgctl(msgid,IPC\_RMID,0)==-1)

{

fprintf(stderr,"msgctl(IPC\_RMID) failed\n");

exit(EXIT\_FAILURE);

}

exit(EXIT\_SUCCESS);

}

**Octal Receiver:**

#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;

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);

}

printf("Decimal to Octal : %s\n",some\_data.some\_text);

if (strncmp(some\_data.some\_text, "end", 3) == 0)

{

running = 0;

printf("Program Terminated\n");

break;

}

}

if(msgctl(msgid,IPC\_RMID,0)==-1)

{

fprintf(stderr,"msgctl(IPC\_RMID) failed\n");

exit(EXIT\_FAILURE);

}

exit(EXIT\_SUCCESS);

}

**Hexadecimal Receiver:**

#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;

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);

}

printf("Decimal to Hexa : %s\n",some\_data.some\_text);

if (strncmp(some\_data.some\_text, "end", 3) == 0)

{

running = 0;

printf("Program Terminated\n");

break;

}

}

if(msgctl(msgid,IPC\_RMID,0)==-1)

{

fprintf(stderr,"msgctl(IPC\_RMID) failed\n");

exit(EXIT\_FAILURE);

}

exit(EXIT\_SUCCESS);

}