**Binary**  
#include <stdio.h>

#include <stdlib.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 decimalToBinary(int num) {

long long int bin = 0;

int place = 1;

while (num > 0) {

int remainder = num % 2;

bin += remainder \* place;

num /= 2;

place \*= 10; }

return bin;

}

int main() {

int running = 1;

struct my\_msg\_st some\_data\_bin;

int msgid;

char buffer[MAX\_TEXT];

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 a decimal number (or 'end' to exit): ")

fgets(buffer, MAX\_TEXT, stdin);

buffer[strcspn(buffer, "\n")] = 0;

if (strncmp(buffer, "end", 3) == 0) {

strcpy(some\_data\_bin.some\_text, "end");

some\_data\_bin.my\_msg\_type = 2;

} else {

int num = atoi(buffer);

int binary = decimalToBinary(num);

snprintf(some\_data\_bin.some\_text, MAX\_TEXT, "%d", binary);

some\_data\_bin.my\_msg\_type = 2;

}

if (msgsnd(msgid, (void\*)&some\_data\_bin, MAX\_TEXT, 0) == -1) {

fprintf(stderr, "msgsnd failed\n");

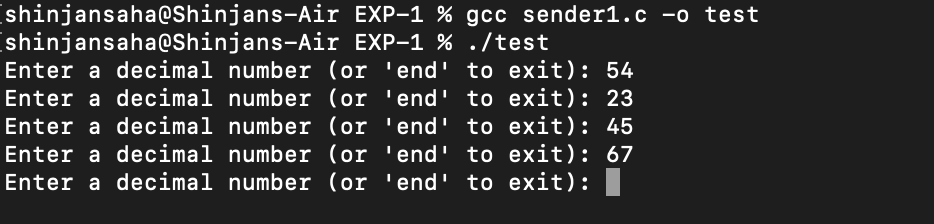
exit(EXIT\_FAILURE);

} if (strncmp(buffer, "end", 3) == 0) {

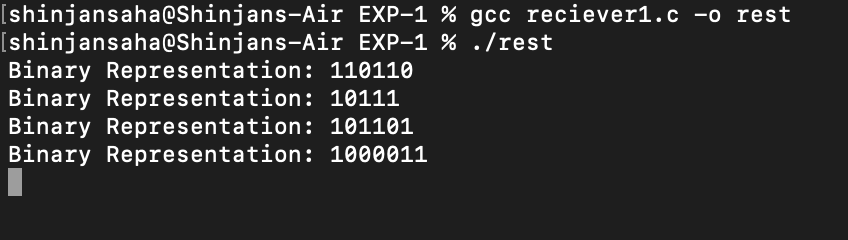
running = 0 } }

exit(EXIT\_SUCCESS);}

**Sender Terminal:-**



**Receiver Terminal:-**



**Octal**

#include <stdio.h>

#include <stdlib.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 decimalToOctal(int num) {

int octal = 0, place = 1;

while (num > 0) {

int remainder = num % 8;

octal += remainder \* place;

num /= 8;

place \*= 10; }

return octal;}

int main() {

int running = 1;

struct my\_msg\_st some\_data\_oct;

int msgid;

char buffer[MAX\_TEXT];

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 a decimal number (or 'end' to exit): ");

fgets(buffer, MAX\_TEXT, stdin);

buffer[strcspn(buffer, "\n")] = 0;

if (strncmp(buffer, "end", 3) == 0) {

strcpy(some\_data\_oct.some\_text, "end");

some\_data\_oct.my\_msg\_type = 2;

} else {

int num = atoi(buffer);

int octal = decimalToOctal(num);

snprintf(some\_data\_oct.some\_text, MAX\_TEXT, "%d", octal);

some\_data\_oct.my\_msg\_type = 2;

}

if (msgsnd(msgid, (void\*)&some\_data\_oct, MAX\_TEXT, 0) == -1) {

fprintf(stderr, "msgsnd failed\n");

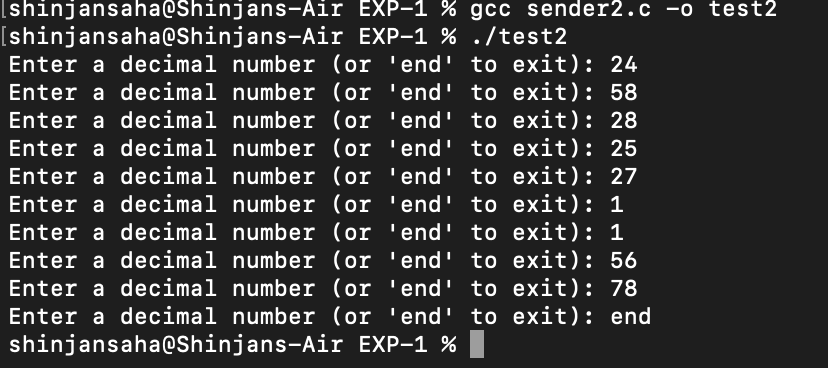
exit(EXIT\_FAILURE);}

if (strncmp(buffer, "end", 3) == 0) {

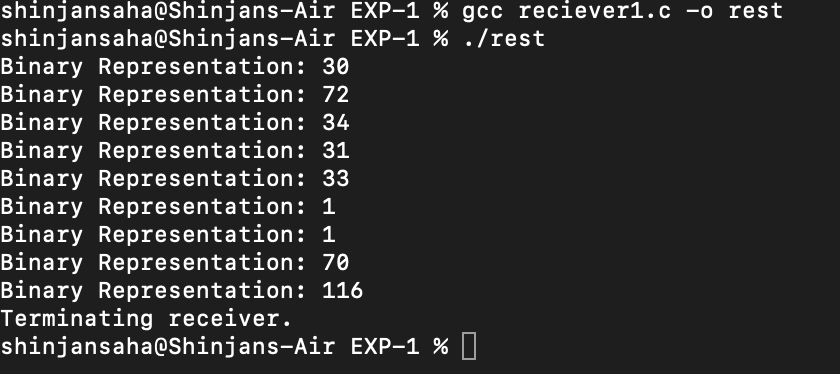
running = 0; }}

exit(EXIT\_SUCCESS);}

**Sender Terminal:-**



**Receiver Terminal:-**



**Hexadecimal**  
#include <stdio.h>

#include <stdlib.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];

};

void decimalToHexadecimal(int num, char \*hexStr) {

snprintf(hexStr, MAX\_TEXT, "%X", num);

}

int main() {

int running = 1;

struct my\_msg\_st some\_data\_hex;

int msgid;

char buffer[MAX\_TEXT];

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 a decimal number (or 'end' to exit): ");

fgets(buffer, MAX\_TEXT, stdin);

buffer[strcspn(buffer, "\n")] = 0;

if (strncmp(buffer, "end", 3) == 0) {

strcpy(some\_data\_hex.some\_text, "end");

some\_data\_hex.my\_msg\_type = 2;

} else {

int num = atoi(buffer);

decimalToHexadecimal(num, some\_data\_hex.some\_text);

some\_data\_hex.my\_msg\_type = 2;

}

if (msgsnd(msgid, (void\*)&some\_data\_hex, MAX\_TEXT, 0) == -1) {

fprintf(stderr, "msgsnd failed\n");

exit(EXIT\_FAILURE);

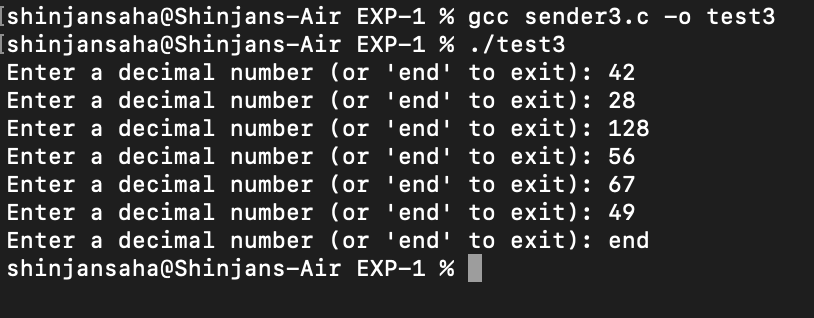
}

if (strncmp(buffer, "end", 3) == 0) {

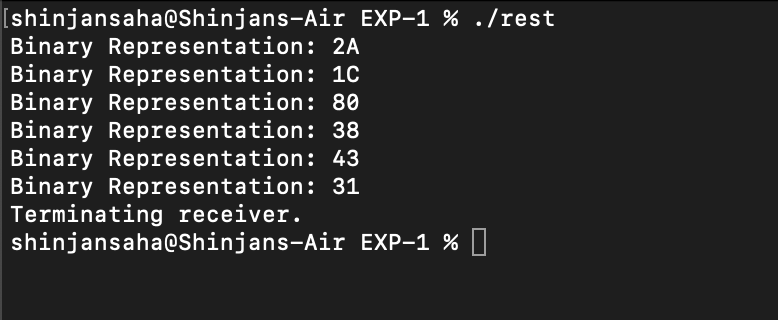
running = 0;}}

exit(EXIT\_SUCCESS);}

**Sender Terminal:-**



**Receiver Terminal:-**



**Receiver**

#include <stdio.h>

#include <stdlib.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, MAX\_TEXT, 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) {

printf("Terminating receiver.\n");

running = 0;

} else {

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

}

}

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

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

exit(EXIT\_FAILURE);

}

exit(EXIT\_SUCCESS);

}

**Binary**

#include <stdio.h>

#include <stdlib.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];

};

void decimalToBinary(int num, char \*binStr) {

if (num == 0) {

strcpy(binStr, "0");

return;}

char temp[MAX\_TEXT] = {0};

int i = 0;

while (num > 0) {

temp[i++] = (num % 2) + '0';

num /= 2; }

temp[i] = '\0';

int len = strlen(temp);

for (int j = 0; j < len; j++) {

binStr[j] = temp[len - j - 1]; }

binStr[len] = '\0';}

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, MAX\_TEXT, 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) {

printf("Terminating receiver.\n");

running = 0;

} else {

int num = atoi(some\_data.some\_text);

char binStr[MAX\_TEXT];

decimalToBinary(num, binStr);

printf("Binary Representation: %s\n", binStr);}

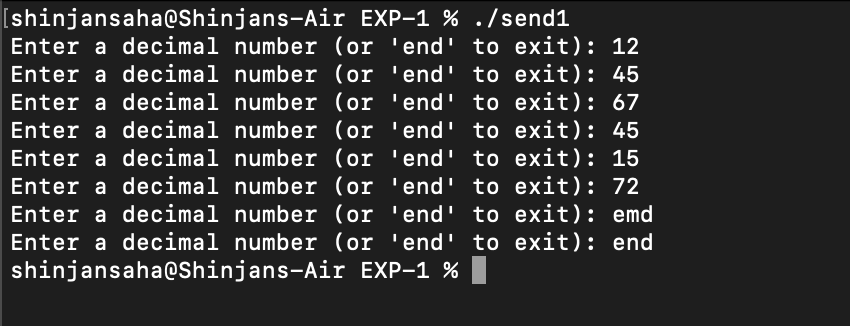
}if (msgctl(msgid, IPC\_RMID, 0) == -1) {

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

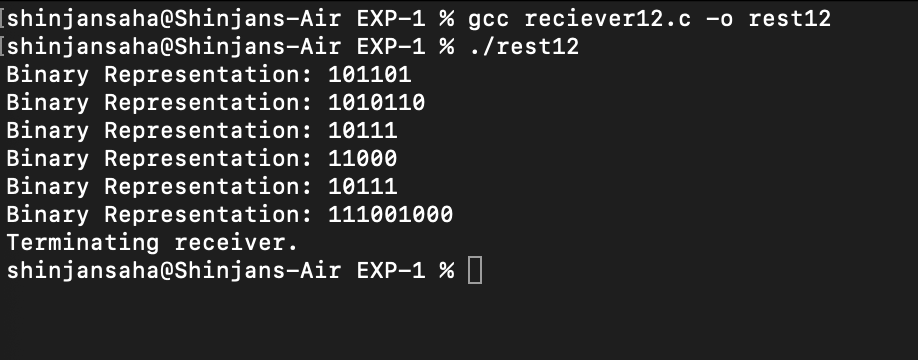
exit(EXIT\_FAILURE);}

exit(EXIT\_SUCCESS);}

**Sender:-**



**Receiver:-**



**Octal**

#include <stdio.h>

#include <stdlib.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];

};

void decimalToOctal(int num, char \*octalStr) {

snprintf(octalStr, MAX\_TEXT, "%o", num);

}

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, MAX\_TEXT, 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) {

printf("Terminating receiver.\n");

running = 0;

} else {

int num = atoi(some\_data.some\_text);

char octalStr[MAX\_TEXT];

decimalToOctal(num, octalStr);

printf("Octal Representation: %s\n", octalStr);

}}

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

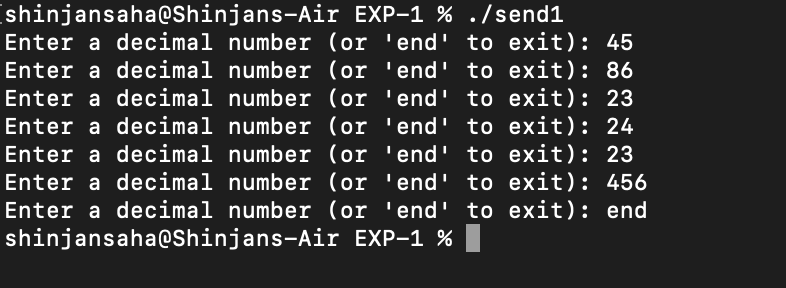
fprintf(stderr, "msgctl(IPC\_RMID) failed\n");

exit(EXIT\_FAILURE);

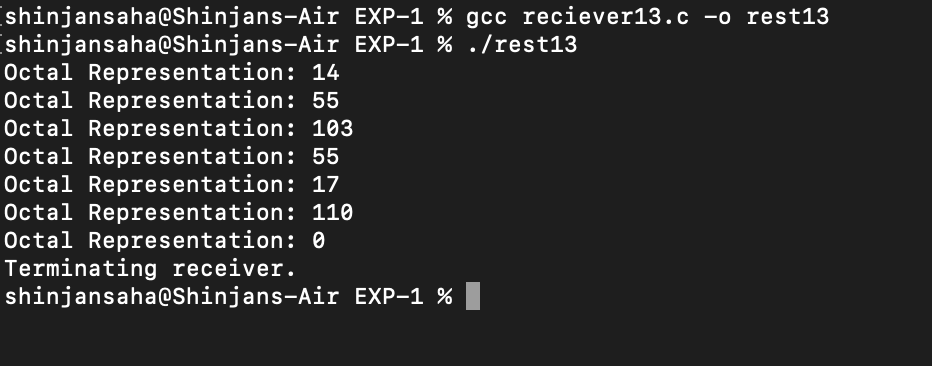
} exit(EXIT\_SUCCESS);

}

**Sender:-**



**Receiver:-**



**Hexadecimal**

#include <stdio.h>

#include <stdlib.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];

};

void decimalToHexadecimal(int num, char \*hexStr) {

snprintf(hexStr, MAX\_TEXT, "%X", num);

}

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, MAX\_TEXT, 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) {

printf("Terminating receiver.\n");

running = 0;

} else {

int num = atoi(some\_data.some\_text);

char hexStr[MAX\_TEXT];

decimalToHexadecimal(num, hexStr);

printf("Hexadecimal Representation: %s\n", hexStr);

}

}

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

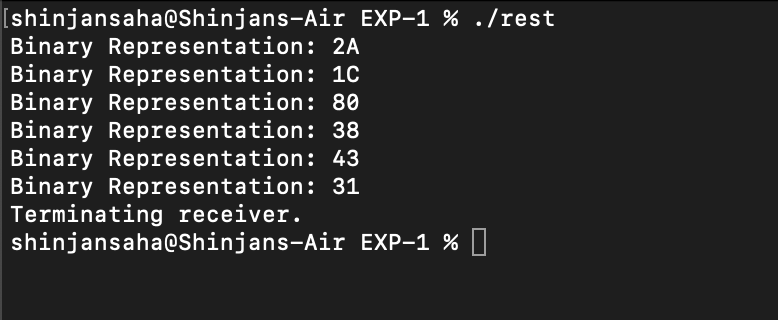
fprintf(stderr, "msgctl(IPC\_RMID) failed\n");

exit(EXIT\_FAILURE);

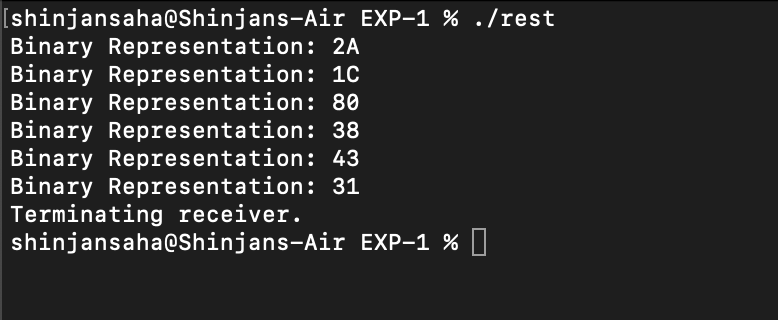
} exit(EXIT\_SUCCESS);

}

**Sender:-**



**Receiver;-**



**Sender:-**

#include <stdio.h>

#include <stdlib.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;

int msgid;

char buffer[MAX\_TEXT];

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 a decimal number (or 'end' to exit): ");

fgets(buffer, MAX\_TEXT, stdin);

buffer[strcspn(buffer, "\n")] = 0;

if (strncmp(buffer, "end", 3) == 0) {

strcpy(some\_data.some\_text, "end");

some\_data.my\_msg\_type = 2;

} else {

strcpy(some\_data.some\_text, buffer);

some\_data.my\_msg\_type = 2;

}

if (msgsnd(msgid, (void\*)&some\_data, MAX\_TEXT, 0) == -1) {

fprintf(stderr, "msgsnd failed\n");

exit(EXIT\_FAILURE);

}

if (strncmp(buffer, "end", 3) == 0) {

running = 0;

}

}

exit(EXIT\_SUCCESS);

}