# Operating System Lab\_02

22CST 蒋云翔

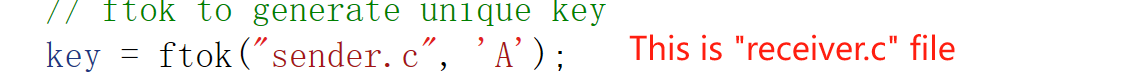
## Task1:

### Idea:

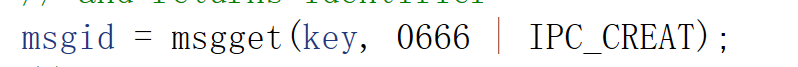
1. In order to access the same queue, the two c program file “sender.c” and “receiver.c” ‘s key value must be the same. So when we call ftok() function, we must two same parameters to the function.

图形用户界面, 应用程序

描述已自动生成



1. The parameter we set in msgget() function is important also, which is mentioned in teacher. He’s lab\_02 file. And 0666 means that users, groups and others can read and writes the message.





文本

描述已自动生成

3. I also do a deletion operation in c file “sender.c”. At first it has a msgctl() function, which destroys the message queue, resulting in a fact that we can not receiver the message from the queue because it has been destroyed (I guess our dear teacher He set this trap to us😄)

图形用户界面, 应用程序, Word

描述已自动生成

4. In my opinion, my code can be optimized more useful and powerful to receiver multiple lines of data, but I fail to make it come true through using while loop. Maybe I will ask u for help.

### Code structure

File “sender.c”

|  |
| --- |
| /\*  \*\* sender.c -- writes to a message queue  \*/  #include <stdio.h>  #include <stdlib.h>  #include <errno.h>  #include <string.h>  #include <sys/types.h>  #include <sys/ipc.h>  #include <sys/msg.h>  struct my\_msgbuf {  long mtype;  char mtext[200];  };  int main(void)  {  struct my\_msgbuf buf;  int msqid;  key\_t key;  if ((key = ftok("sender.c", 'A')) == -1) {  perror("ftok");  exit(1);  }  if ((msqid = msgget(key, 0666 | IPC\_CREAT)) == -1) {  perror("msgget");  exit(1);  }    printf("Enter lines of text, ^D to quit:\n");  buf.mtype = 1; /\* we don't really care in this case \*/  while(fgets(buf.mtext, sizeof buf.mtext, stdin) != NULL) {  int len = strlen(buf.mtext);  /\* ditch newline at end, if it exists \*/  if (buf.mtext[len-1] == '\n') buf.mtext[len-1] = '\0';  if (msgsnd(msqid, &buf, len+1, 0) == -1) /\* +1 for '\0' \*/  perror("msgsnd");  }  return 0;  } |

File “receiver.c”

|  |
| --- |
| // C Program for Message Queue (Reader Process)  #include <stdio.h>  #include <sys/ipc.h>  #include <sys/msg.h>  // structure for message queue  struct mesg\_buffer {  long mesg\_type;  char mesg\_text[200];  } message;  int main()  {  key\_t key;  int msgid;  // ftok to generate unique key  key = ftok("sender.c", 'A');  // msgget creates a message queue  // and returns identifier  msgid = msgget(key, 0666 | IPC\_CREAT);  // msgrcv to receive message  msgrcv(msgid, &message, sizeof(message), 1, 0);  // display the message  printf("Data Received is : %s \n",  message.mesg\_text);  // to destroy the message queue  msgctl(msgid, IPC\_RMID, NULL);  return 0;  } |

### Execution results

|  |
| --- |
|  |