운영체제 11주차, 12주차

20135151 이갑성

operating\_systems\_lab09-1\_task

|  |
| --- |
| <소스코드>  [lab8\_1.c]  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <fcntl.h>  #include <sys/types.h>  #include <sys/stat.h>  #define SIZE 128  #define FIFO "fifo"  int main(int argc, char \*argv[])  {  int fd;  char buffer[SIZE];  //FIFO를 생성해준다.  if(mkfifo(FIFO, 0666) == -1){  perror("mkfifo failed");  exit(1);  }  //읽기와 쓰기용으로 FIFO파일을 열어준다.  if((fd = open(FIFO, O\_RDWR)) == -1){  perror("open failed");  exit(1);  }  //FIFO파일을 SIZE크기 만큼 읽어와 buffer에 저장한다.  while(!0){  if(read(fd, buffer, SIZE) == -1){  perror("read failed");  exit(1);  }  if(!strcmp(buffer, "quit")){ //읽어 들어온 값이 quit면 반복문 종료  exit(0);  }  printf("receive message %s \n", buffer); //읽어 들어온 값이 buffer에 저장되어있고  //그 buffer을 출력해준다.  }  }  [lab8\_2.c]  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <fcntl.h>  #include <sys/types.h>  #include <sys/stat.h>  #define SIZE 128  #define FIFO "fifo"  int main(int argc, char \*argv[])  {  int fd, i;  char buffer[SIZE];  //쓰기 전용으로 FIFO파일 열어준다.  if((fd = open(FIFO, O\_WRONLY)) == -1){  perror("open failed");  exit(1);  }  //전달된 인자의 개수만큼 for문을 돌려준다.  //argv에 전달된 값을 buffer에 저장해주고 그 buffer을 FIFO파일에 써준다.  for(i = 1; i < argc; i++){  strcpy(buffer, argv[i]);  if(write(fd, buffer, SIZE) == -1){  perror("write failed");  exit(1);  }  }  exit(0);  } |
| <실행결과> |

operating\_systems\_lab09-2\_task

|  |
| --- |
| <소스코드>  [lab10\_3.c]  #include <stdio.h>  #include <string.h>  #include <sys/types.h>  #include <sys/ipc.h>  #include <sys/msg.h>  #include <stdlib.h>  #define SIZE 1024  //메세지 보낼 데이터를 담는 구조체  struct {  long type;  char data[SIZE];  } msg\_data;  int main()  {  int msqid, data\_len;  char buffer[SIZE];    //lab10\_3의 프로세스는 msgget()으로 키가 20135151인 메세지 큐를 생성한다.  if((msqid = msgget((key\_t)20135151, IPC\_CREAT|0666)) == -1){  perror("msgget failed");  exit(1);  }  //입력 데이터 값을 buffer로 받아주고 입력한 데이터의 순위를 입력받는다.  //버퍼에 있는 데이터 값을 구조체 msg\_data의 data에 대입해준다.  //msgsnd함수로 msg\_data를 메세지 큐에 전송해준다.  while(!0){  printf("input data => ");  scanf(" %[^\n]", buffer);  //입력 버퍼값이 quit일때는 무한반복문을 종료시켜준다.  if(!strcmp(buffer, "quit")){  break;  }  printf("input priority => ");  scanf("%ld", &(msg\_data.type));  strcpy(msg\_data.data, buffer);  if((msgsnd(msqid, &msg\_data, strlen(msg\_data.data), 0)) == -1){  perror("msgsnd failed");  exit(1);  }  }  exit(0);  }  [lab10\_4.c]  #include <stdio.h>  #include <sys/types.h>  #include <sys/ipc.h>  #include <sys/msg.h>  #include <errno.h>  #include <stdlib.h>  #define SIZE 1024  #define PRIOR 10  struct {  long type;  char data[SIZE];  } msg\_data;  int main()  {  int msqid, data\_len;  //msgget함수를 이용해 키가 20135151인 메시지 큐로 접근한다.  if((msqid = msgget((key\_t)20135151, IPC\_CREAT|0666)) == -1){  perror("msgget failed");  exit(1);  }  //msgrcv함수로 메세지를 읽어들어오는데 수신할 메시지 선택 조건이 -10임으로  //-10의 절대값이 10보다 작거나 같은 메세지 형식을 갖는 메세지를 모두 해당한다.  //msgrcv함수의 반환값은 성공시 수신한 메시지 크기임으로 성공시 if문 조건문에 true에 해당한다.  //  while(!0){  if((data\_len = msgrcv(msqid, &msg\_data, SIZE, (-1\*PRIOR), IPC\_NOWAIT)) == -1){  if(errno == ENOMSG){ //에러발생이 ENOMSG일때 해당 문장 출력해준다.  printf("no more messages \n");  break;  }  perror("msgrcv failed");  break;  }  msg\_data.data[data\_len] = '\0'; //해당 데이터에 마지막에 \0을 넣어줌으로써 문자열로 변환  printf("data: %s [%ld]\n", msg\_data.data, msg\_data.type); //메세지 정보 출력  }  //msgctl함수를 이용해서 메세지 큐를 제거해준다.  if(msgctl(msqid, IPC\_RMID, 0) == -1){  perror("msgctl failed");  exit(1);  }  exit(0);  } |
| <실행결과> |