운영체제

20135151 이갑성

|  |  |  |  |
| --- | --- | --- | --- |
| [과제 1]  #include <stdio.h>  #include <signal.h>  #include <unistd.h>  #include <stdlib.h>  void signal\_handler(int signo);  int main()  {  sigset\_t set, oldset;  struct sigaction act;  act.sa\_handler = signal\_handler; // 시그널을 받았을때 signal\_handler 함수를 취함  sigemptyset(&act.sa\_mask); // //act.sa\_mask 시그널 집합을 비워서 시그널을 처리하는 동안 발생한 모든 시그널을 블록화 시키지 않는다  act.sa\_flags = 0;  sigaction(SIGINT, &act, NULL); //SIGINT 시그널에 대해서 새롭게 act로 설정해준다.  sigemptyset(&set); //set 시그널 집합을 생성해준다.  sigaddset(&set, SIGINT); //set 시그널 집합에 SIGINT 시그널을 추가한다.  sigprocmask(SIG\_BLOCK, &set, &oldset); //기존에 블록화된 시그널 집합에 set 시그널 집합을 추가  sleep(10);  sigprocmask(SIG\_SETMASK, &oldset, NULL); //기존에 블록화된 시그널 집합에 oldset 시그널 집합으로 교체한다.  pause(); //실행을 중단킨다(시그널이 도착할때까지)  exit(0);  }  void signal\_handler(int signo)  {  printf("I'm your signal %d \n", signo);  }   |  | | --- | | Case1)  #include <stdio.h>  #include <signal.h>  #include <unistd.h>  #include <stdlib.h>  void signal\_handler(int signo);  int main()  {  sigset\_t set, oldset;  struct sigaction act;  act.sa\_handler = signal\_handler; // 시그널을 받았을때 signal\_handler 함수를 취함  sigemptyset(&act.sa\_mask); // //act.sa\_mask 시그널 집합을 비워서 시그널을 처리하는 동안 발생한 모든 시그널을 블록화 시키지 않는다  act.sa\_flags = 0;  sigaction(SIGINT, &act, NULL); //SIGINT 시그널에 대해서 새롭게 act로 설정해준다.  sigemptyset(&set); //set 시그널 집합을 생성해준다.  sigaddset(&set, SIGINT); //set 시그널 집합에 SIGINT 시그널을 추가한다.  sigprocmask(SIG\_BLOCK, &set, &oldset); //기존에 블록화된 시그널 집합에 set 시그널 집합을 추가  sleep(10);  sigprocmask(SIG\_SETMASK, &oldset, NULL); //기존에 블록화된 시그널 집합에 oldset 시그널 집합으로 교체한다.  pause(); //실행을 중단킨다(시그널이 도착할때까지)  exit(0);  }  void signal\_handler(int signo)  {  printf("I'm your signal %d \n", signo);  }  **해설: Sleep(10) 뒤에 SIGINT를 블록화에서 해제해줌으로써 10초 뒤에 Ctrl-C키에 의해 SIGINT가 처리된다.** | | Case2)  #include <stdio.h>  #include <signal.h>  #include <unistd.h>  #include <stdlib.h>  void signal\_handler(int signo);  int main()  {  sigset\_t set, oldset;  struct sigaction act;  act.sa\_handler = signal\_handler; // 시그널을 받았을때 signal\_handler 함수를 취함  sigemptyset(&act.sa\_mask); // //act.sa\_mask 시그널 집합을 비워서 시그널을 처리하는 동안 발생한 모든 시그널을 블록화 시키지 않는다  act.sa\_flags = 0;  sigaction(SIGINT, &act, NULL); //SIGINT 시그널에 대해서 새롭게 act로 설정해준다.  sigemptyset(&set); //set 시그널 집합을 생성해준다.  sigaddset(&set, SIGINT); //set 시그널 집합에 SIGINT 시그널을 추가한다.  sigprocmask(SIG\_BLOCK, &set, &oldset); //기존에 블록화된 시그널 집합에 set 시그널 집합을 추가  sigprocmask(SIG\_SETMASK, &oldset, NULL); //기존에 블록화된 시그널 집합에 oldset 시그널 집합으로 교체한다.  sleep(10);  pause(); //실행을 중단킨다(시그널이 도착할때까지)  exit(0);  }  void signal\_handler(int signo)  {  printf("I'm your signal %d \n", signo);  }  해설: **Sleep(10) 앞에 SIGINT를 블록화에서 해제해줌으로써 시간 경과에 상관없이 Ctrl-C키에 의해 SIGINT가 처리된다.** | | Case3)  #include <stdio.h>  #include <signal.h>  #include <unistd.h>  #include <stdlib.h>  void signal\_handler(int signo);  int main()  {  sigset\_t set, oldset;  struct sigaction act;  act.sa\_handler = signal\_handler; // 시그널을 받았을때 signal\_handler 함수를 취함  sigemptyset(&act.sa\_mask); //시그 집합을 비워서 시그널을 처리하는 동안 발생한 모든 시그널을  //블록화 시키지 않는다.  act.sa\_flags = 0;  sigaction(SIGINT, &act, NULL); //SIGINT 시그널에 대해서 새롭게 act 설정해준다.  sigemptyset(&set); //set 시그널 집합을 생성해준다.  sigaddset(&set, SIGINT); //set 시그널 집합에 SIGINT 시그널을 추가한다.    sigprocmask(SIG\_BLOCK, &set, &oldset); //기존에 블록화된 시그널 집합에 set 시그널 집합을 추가    sleep(10);    pause(); //실행을 중단킨다(시그널이 도착할때까지)    sigprocmask(SIG\_SETMASK, &oldset, NULL); //기존에 블록화된 시그널 집합에 oldset 시그널 집합으로 교체한다.    exit(0);  }  void signal\_handler(int signo)  {  printf("I'm your signal %d \n", signo);  }  **해설: pause()뒤에 SIGINT를 블록화를 해제하는 것을 설정해서**  **Ctr-C 키를 눌러도 pause()때문에 대기상태에 빠짐으로, SIGINT가 블록화 상태가 유지되어 처리가 안되는 상황이된다.** | |

|  |
| --- |
| [과제2]  #include <stdio.h>  #include <signal.h>  #include <unistd.h>  #include <stdlib.h>  void signal\_handler(int signo);  int main()  {  sigset\_t set, oldset;  struct sigaction act;  act.sa\_handler = signal\_handler; //시그널 받았을때 signal\_handler 함수를 취함  sigemptyset(&act.sa\_mask); //act.sa\_mask 시그널 집합을 비워서 시그널을 처리하는 동안 발생한 >모든 시그널을 블록화 시키지 않는다  act.sa\_flags = 0;  sigaction(SIGINT, &act, NULL); //SIGINT 시그널에 대해서 새롭게 act로 설정해준다.  sigemptyset(&set); //set집합을 비워준다.  sigaddset(&set, SIGINT); //set집합에 SIGINT 시그널을 넣어준다.  sigprocmask(SIG\_BLOCK, &set, &oldset); //기존에 블록화된 시그널 집합에 set시그널 집합을 추가  sleep(10);  sigsuspend(&oldset); //oldset 시그널 집합을 블록을 설정해줌과 동시에 시그널이 도착할 때까지 중단해준다.  exit(0);  }  void signal\_handler(int signo)  {  printf("I'm your signal %d \n", signo);  }  /\*  이유: 과제1에서는 sigprocmask() 호출과 pause() 호출 사이에서도 시그널이 발생할 수 있는데  이 때 발생한 시그널을 잃어 버릴 수 있다고 한다. 그러므로 블록 설정과 동시에 시그널 도착을  바로 확인해야 시그널을 잃어 버리는 일이 없다. sigsuspend()함수는 블록 설정과 시그널을 대기하는 것을 동시에 하기때문에 한번만 Ctrl-C 누르면 종료되는 것이다.  \*/  ~ |