1. 소스코드

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <errno.h>  #include <sys/resource.h>  #include <sys/wait.h>  double ssu\_maketime(struct timeval \*time);  void term\_stat(int stat);  void ssu\_print\_child\_info(int stat, struct rusage \*rusage);  int main(void) {  struct rusage rusage;  pid\_t pid;  int status;  if((pid = fork()) == 0) {//자식 프로세스 실행  //execv() 인자로 넘겨줄 배열 초기화  char \*args[] = {"find", "/", "-maxdepth", "4", "-name", "stdio.h", NULL};  //자식 프로세스를 대신하여 새로운 프로세스 실행  if(execv("/usr/bin/find", args) < 0) {  fprintf(stderr, "exec error\n");  exit(1);  }  }  //종료된 프로세스의 자원 사용 정보를 rusage 구조체 변수를 통해 받음  if(wait3(&status, 0, &rusage) == pid)  ssu\_print\_child\_info(status, &rusage);  else {  fprintf(stderr, "wait3 error\n");  exit(1);  }  exit(0);  }  //시간 계산  double ssu\_maketime(struct timeval \*time) {  return ((double)time -> tv\_sec + (double)time -> tv\_usec / 1000000.0);  }  void term\_stat(int stat) {  if(WIFEXITED(stat))//자식 프로세스가 정상 종료 시  printf("normally terminated. exit status = %d\n", WEXITSTATUS(stat));//exit() 인수 하위 8비트 값 출력  else if(WIFSIGNALED(stat))//자식 프로세스가 시그널을 받았으나 비정상적으로 종료가 되었다면  printf("abnormal termination by signal %d. %s\n", WTERMSIG(stat),//시그널 번호 반환  #ifdef WCOREDUMP//코어 파일이 생성된 경우  WCOREDUMP(stat) ? "core dumped":"no core"  #else  NULL  #endif  );  else if(WIFSTOPPED(stat))//자식 프로세스가 현재 중지 상태인 경우  printf("stopped by signal %d\n", WSTOPSIG(stat));  }  //자식 프로세스 정보 출력  void ssu\_print\_child\_info(int stat, struct rusage \*rusage) {  printf("Termination info follows\n");  term\_stat(stat);  printf("user CPU time : %.2f(sec)\n",  ssu\_maketime(&rusage->ru\_utime));  printf("system CPU time : %.2f(sec)\n",  ssu\_maketime(&rusage->ru\_stime));  } |

2. 실행결과

