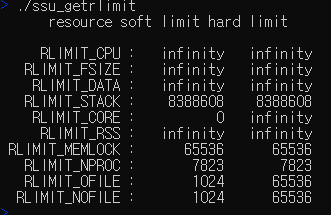
컴퓨터학부 20162448 김병준

1. 결과



1. 소스코드

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <string.h>  #include <sys/resource.h>  #define ssu\_show(limit) ssu\_show\_limit(limit, #limit)  void ssu\_show\_limit(int rlim\_type, char \*rlim\_name);  int main(void)  {  // 프로세스마다 적용된 자원 한계 값 출력  printf(" resource soft limit hard limit \n\n");  ssu\_show(RLIMIT\_CPU); // 프로세스가 소비할 수 있는 최대 CPU 시간  ssu\_show(RLIMIT\_FSIZE); // 파일 생성 시 파일의 최대 크기(바이트)  ssu\_show(RLIMIT\_DATA); // 자료 구역 전체의 최대 크기(바이트)  ssu\_show(RLIMIT\_STACK); // 스택 최대 크기(바이트)  ssu\_show(RLIMIT\_CORE); // 한 코어 파일의 최대 크기(바이트)  #ifdef RLIMIT\_RSS // 메모리 상주 세트 크기(Resident Set Size)의 최댓값(바이트)  ssu\_show(RLIMIT\_RSS);  #endif  #ifdef RLIMIT\_MEMLOCK // 한 프로세스가 mlock으로 잠글 수 있는 메모리의 최대 용량(바이트)  ssu\_show(RLIMIT\_MEMLOCK);  #endif  #ifdef RLIMIT\_NPROC // 실제 사용자 ID당 최대 자식 프로세스 개수  ssu\_show(RLIMIT\_NPROC);  #endif  #ifdef RLIMIT\_OFILE // RLIMIT\_NOFILE에 대한 BSD 이름  ssu\_show(RLIMIT\_OFILE);  #endif  #ifdef RLIMIT\_NOFILE // 한 프로세스가 열어 둘 수 있는 최대 파일 개수  ssu\_show(RLIMIT\_NOFILE);  #endif  #ifdef RLIMIT\_VMEN // RLIMIT\_AS와 동일, 한 프로세스의 총 자유 메모리 용량의 최댓값(바이트)  ssu\_show(RLIMIT\_VMEN);  #endif  exit(0);  }  /\*  struct rlimit {  rlim\_t rlim\_cur; // 약한 한계  rlim\_t rlim\_max; // 강한 한계  }  \*/  void ssu\_show\_limit(int rlim\_type, char \*rlim\_name)  {  struct rlimit rlim;  char cur[11], max[11];  getrlimit(rlim\_type, &rlim);  if(rlim.rlim\_cur == RLIM\_INFINITY) // 현재 잡힌 크기가 무한대의 경우  strcpy(cur, "infinity");  else  sprintf(cur, "%10ld", rlim.rlim\_cur);  if(rlim.rlim\_max == RLIM\_INFINITY) // 최대 크기가 무한대의 경우  strcpy(max, "infinity");  else  sprintf(max, "%10ld", rlim.rlim\_max);  printf("%15s : %10s %10s\n", rlim\_name, cur, max); // 출력  } |