컴퓨터학부 20201841 박세연

<setbuf1>

1. 소스코드

**[ssu\_setbuf\_1.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#define BUFFER\_SIZE 1024

int main(void){

char buf[BUFFER\_SIZE];

setbuf(stdout, buf);

// setbuf 함수를 이용하여 stdout의 버퍼를 위에서 정의한 buf로 설정

// stdout의 출력이 즉시 화면에 나타나는 것이 아니라 먼저 buf 배열에 저장되게 됨

// 이 배열이 꽉 차거나 버퍼가 플러시 될 때까지 출력 내용이 화면에 나타나지 않음

printf("Hello, ");

sleep(1);

printf("OSLAB!!");

sleep(1);

printf("\n");

sleep(1);

// 위의 세 출력은 사용자 정의 버퍼를 사용하여 출력됨

setbuf(stdout, NULL);

// setbuf 함수를 이용하여 버퍼를 NULL로 설정하면 버퍼링 없이 직접 출력됨

printf("How");

sleep(1);

printf(" are");

sleep(1);

printf(" you?");

sleep(1);

printf("\n");

exit(0);

}

1. 실행결과

텍스트, 소프트웨어, 멀티미디어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

<setbuf2>

1. 소스코드

**[ssu\_setbuf\_2.c]**

#include <stdio.h>

#include <stdlib.h>

#define BUFFER\_SIZE 1024

int main(void){

char buf[BUFFER\_SIZE];

int a,b;

int i;

// setbuf 함수로 stdin의 버퍼를 buf 배열로 설정함

// 표준 입력으로부터 읽어들인 데이터가 우선적으로 buf 배열에 저장됨

setbuf(stdin, buf);

scanf("%d %d", &a, &b);

// 입력된 버퍼의 내용을 순회하며 출력

for (i = 0; buf[i] != '\n'; i++)

putchar(buf[i]);

putchar('\n');

exit(0);

}

2. 실행결과

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

<setvbuf>

1. 소스코드

**[ssu\_setvbuf.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#define BUFFER\_SIZE 1024

void ssu\_setbuf(FILE \*fp, char \*buf);

int main(void){

char buf[BUFFER\_SIZE];

char \*fname = "/dev/pts/2";

// tty 명령어로부터 현재 표준 입력에 접속된 터미널 장치 파일 이름을 알아냄

FILE \*fp;

// 현재 버퍼 오픈

if ((fp = fopen(fname, "w")) == NULL) {

fprintf(stderr, "fopen error for %s", fname);

exit(1);

}

ssu\_setbuf(fp, buf);

// ssu\_setbuf 함수를 호출하여 파일 스트림의 버퍼링을 설정

// 여기서는 사용자 정의 버퍼인 buf를 사용

fprintf(fp, "Hello, ");

sleep(1);

fprintf(fp, "UNIX!!");

sleep(1);

fprintf(fp, "\n");

sleep(1);

// 버퍼가 설정된 후 실행되기 때문에 버퍼에 넣은 후 한 번에 출력됨

ssu\_setbuf(fp, NULL);

// 여기서는 NULL을 전달하여 비버퍼링 모드로 전환

fprintf(fp, "HOW");

sleep(1);

fprintf(fp, " ARE");

sleep(1);

fprintf(fp, " YOU?");

sleep(1);

fprintf(fp, "\n");

sleep(1);

// fprintf()를 호출할때마다 버퍼에 넣지 않고 바로 출력

exit(0);

}

void ssu\_setbuf(FILE \*fp, char \*buf) {

size\_t size;

int fd;

int mode;

fd = fileno(fp); // 파일 스트림에서 사용 중인 파일 디스크립터 얻음

if (isatty(fd)) // 해당 파일 디스크립터가 터미널에 연결되어있는지 확인

mode = \_IOLBF; // 연결되어있다면 라인버퍼링모드를 사용

else

mode = \_IONBF; // 그렇지 않다면 비버퍼링모드를 사용

// 제공된 버퍼인 buf가 NULL이면 비버퍼링 모드 선택

if (buf == NULL) {

mode = \_IONBF;

size = 0;

}else

size = BUFFER\_SIZE; // 그렇지 않으면 BUFFER\_SIZE 만큼의 버퍼 크기를 설정

setvbuf(fp, buf, mode, size);

// setvbuf로 파일 스트림의 버퍼링 설정을 적용

}

2. 실행결과

텍스트, 스크린샷, 멀티미디어, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

<\_exit(), exit()>

1. 소스코드

**[ssu\_\_exit.c]**

#include <stdio.h>

#include <stdlib.h>

int main(void){

printf("Good afternoon?");

\_exit(0);

// \_exit() 함수를 호출하여 강제종료 -> 이때 표준 입출력이 다 날라감

// 출력 X

}

**[ssu\_exit.c]**

#include <stdio.h>

#include <stdlib.h>

int main(void){

printf("Good afternoon");

exit(0);

// exit() 함수를 호출하여 강제 종료 -> 표준 입출력 처리를 모두 마치고 \_exit()를 부르기 때문에

// 출력 O

}

2. 실행결과

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

<atexit1>

1. 소스코드

**[ssu\_atexit\_2.c]**

#include <stdio.h>

#include <stdlib.h>

void ssu\_out(void);

int main(void){

if (atexit(ssu\_out)) {

// atexit() 함수를 사용해 exit() 실행 시 자동으로 호출 될 함수를 등록함

// 인자로 ssu\_out을 넘겨주었기 때문에 exit() 실행 시 ssu\_out 함수를 호출함

fprintf(stderr, "atexit error\n");

exit(1);

}

exit(0);

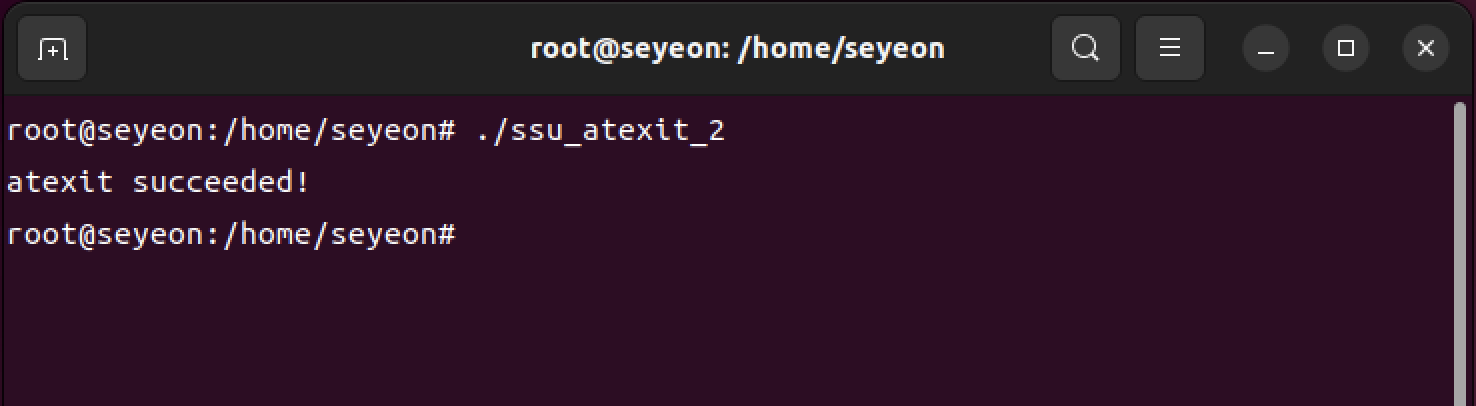
}

void ssu\_out(void) {

printf("atexit succeeded!\n");

}

2. 실행결과



<atexit2>

1. 소스코드

**[ssu\_atexit\_1.c]**

#include <stdio.h>

#include <stdlib.h>

static void ssu\_exit1(void);

static void ssu\_exit2(void);

int main(void){

if (atexit(ssu\_exit2) != 0) { // ssu\_exit2를 종료 핸들러로 등록

fprintf(stderr, "atexit error for ssu\_exit2");

exit(1);

}

if (atexit(ssu\_exit1) != 0) { // ssu\_exit1을 종료 핸들러로 등록

fprintf(stderr, "atexit error for ssu\_exit1");

exit(1);

}

if (atexit(ssu\_exit1) != 0) { // ssu\_exit1을 종료 핸들러로 등록

fprintf(stderr, "atexit error for ssu\_exit1");

exit(1);

}

printf("done\n"); // 함수 종료 전 done 출력

exit(0); // 정상 종료하며 atexit()에 등록된 함수들이 호출됨

// 프로그램이 종료될 때 atexit()에 의해 등록된 함수들이 역순으로 호출 됨

}

static void ssu\_exit1(void) {

printf("ssu\_exit1 handler\n");

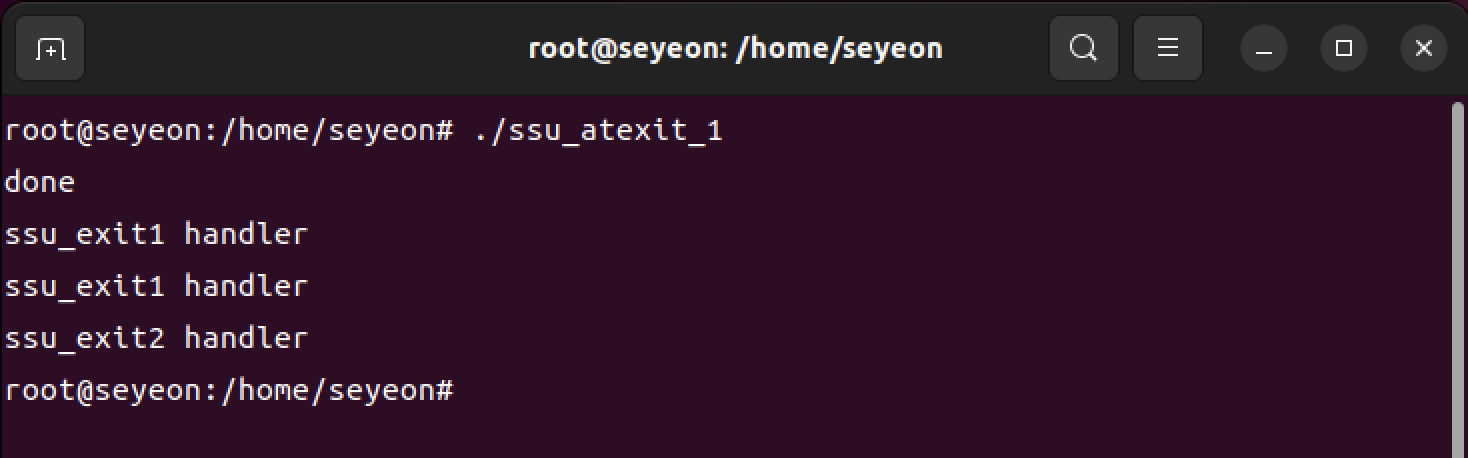
}

static void ssu\_exit2(void) {

printf("ssu\_exit2 handler\n");

}

2. 실행결과



<Command-Line>

1. 소스코드

**[ssu\_command-line.c]**

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char \*argv[]) {

int i;

for (i = 0; i < argc; i++) {

printf("argv[%d] : %s\n", i, argv[i]);

// 프로그램 실행 시에 인자로 받는 값들을 출력함

}

exit(0);

}

2. 실행결과

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명