컴퓨터학부 20202920 조민혁

1.ssu\_setbuf\_1.c의 소스코드와 실행결과

#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 통해서 버퍼를 설정해준다.

printf("Hello, ");

sleep(1);

//hello를 출력하고 1초를 쉰다.

printf("OSLAB!!");

sleep(1);

//OSLAB을 출력하고 1초 쉰다.

printf("\n");

sleep(1);

setbuf(stdout,NULL);

//setbuf에 stdout의 버퍼를 NULL로 설정하여 기존에 설정된 버퍼를 해제한다.

printf("How");

sleep(1);

printf(" are");

sleep(1);

printf(" you?");

sleep(1);

printf("\n");

exit(0);

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2.ssu\_setbuf\_2.c의 소스코드와 실행결과

#include <stdio.h>

#include <stdlib.h>

#define BUFFER\_SIZE 1024

int main(void){

char buf[BUFFER\_SIZE];

int a, b;

int i;

setbuf(stdin,buf);

//setbuf를 통해 버퍼를 설정한다.

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

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

putchar(buf[i]);

//개행문자가 입력될 때까지 입력을 받는다.

putchar('\n');

exit(0);

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

3.ssu\_setvbuf.c의 소스코드와 실행결과

#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/0";

FILE \*fp;

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

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

exit(1);

}//읽기 모드로 파일을 열어준다.

ssu\_setbuf(fp,buf);

fprintf(fp,"Hello, ");

//파일에 Hello 를 적어준다.

sleep(1);

fprintf(fp,"UNIX!!");

sleep(1);

fprintf(fp,"\n");

sleep(1);

ssu\_setbuf(fp,NULL);

//ssu\_setbuf함수를 호출해준다.

fprintf(fp,"HOW");

sleep(1);

fprintf(fp," ARE");

sleep(1);

fprintf(fp," YOU?");

sleep(1);

fprintf(fp, "\n");

sleep(1);

exit(0);

}

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

size\_t size;

int fd;

int mode;

fd = fileno(fp);

//fileno함수를 통해 파일 디스크립터 값을 fd에 저장한다.

if(isatty(fd))

mode = \_IOLBF;

//isatty함수를 통해 파일 디스크립터가 터미널 디바이스인지 확인한다.

else

mode - \_IOFBF;

if(buf == NULL){

mode = \_IONBF;

size = 0;

}

else

size = BUFFER\_SIZE;

//buf의 상태에 따라 if문을 분기해준다

setvbuf(fp,buf,mode,size);

//setvbuf함수를 사용한다.

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

4.ssu\_\_exit.c과 ssu\_exit.c의 소스코드와 실행결과

ssu\_\_exit.c의 소스코드

#include <stdio.h>

#include <unistd.h>

int main(void)

{

printf("Good afternoon?");

\_exit(0);

//\_exit(0)함수를 통해서 프로그램을 종료한다.

}

ssu\_exit.c의 소스코드

#include <stdio.h>

#include <stdlib.h>

int main(void)

{

printf("Good afternoon");

exit(0);

//exit 함수로 프로그램을 종료한다.

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

5.ssu\_atexit\_1.c의 소스코드와 실행결과

#include <stdio.h>

#include <stdlib.h>

static void ssu\_exit1(void);

static void ssu\_exit2(void);

int main(void){

if(atexit(ssu\_exit2) != 0){

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

exit(1);

}

//atexit으로 ssu\_exit2를 설정해줬다.

if(atexit(ssu\_exit1) != 0){

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

exit(1);

}

//다시 한 번 atexit으로 ssu\_exit1을 exit한 후에 실행될 함수로 설정해줬다.

if(atexit(ssu\_exit1) != 0){

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

exit(1);

}

//같은 과정을 한 번 더 반복

printf("done\n");

exit(0);

}

static void ssu\_exit1(void){

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

}

static void ssu\_exit2(void){

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

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

6.ssu\_atexit\_2.c의 소스코드와 실행결과

#include <stdio.h>

#include <stdlib.h>

void ssu\_out(void);

int main(void){

if(atexit(ssu\_out)){

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

exit(1);

}

//atexit을 통해 메인 함수가 리턴한 후 실행할 함수를 등록한다.

exit(0);

}

void ssu\_out(void){

printf("atexit succeeded!\n");

//exit함수가 실행되고 나서 실행될 함수를 만들어준 것이다.

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

7.ssu\_command-line.c의 소스코드와 실행결과

#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]);

//함수의 인자로 들어온 것에 대해서 하나씩 구분하여 출력해준다

exit(0);

}

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명