# prob1 stack frame 그리기

텍스트이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

#prob1,2,3,4

**#prob1**

#include<stdio.h>

int main(){

int a,b;

a=printf("input->");

scanf("%d",&b);

printf("%d:%d",a,b);

}

**#prob2**

#include<stdio.h>

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

if(3!=argc){

printf("%s [Parameter] [Parameter]\n",argv[0]);

}

else{

printf("%s %s %s\n",argv[0],argv[1],argv[2]);

}

}

**#prob3**

#include<stdio.h>

int main(){

int a=0,b=0;

while(b<=99){

a=a+b;

b=b+1;

}

printf("%d\n",a);

return 0;

}

**#prob4**

#include<stdio.h>

#include<time.h>

#include<stdlib.h>

int main(){

int t;

srand(time(NULL));

t=rand();

printf("random value: %d\n",t);

if(t & 1){

printf("Yes!");

}

else{

printf("nono~");

}

}

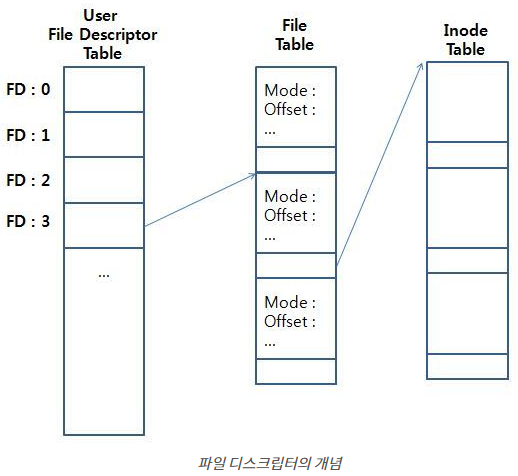
#파일 디스크립터 조사 및 실습

파일 디스크립터: 리눅스, 유닉스 계열 시스템에서 프로세스가 파일을 다룰 때 사용하는 개념으로 프로세스에서 특정 파일에 접근할 때 사용하는 추상적인 값. (일반적으로 0이 아닌 정수)

텍스트이(가) 표시된 사진

자동 생성된 설명

파일을 open하면 커널은 해당 프로세스 파일 디스크립터 숫자 중 사용하지 않는 가장 작은 값을 할당 -> 프로세스가 열려있는 파일에 접근할때 디스크립터 값을 이용해서 파일을 지칭



-실습

#include<stdio.h>

#include<fcntl.h>

#include<unistd.h>

#include<sys/socket.h>

int main(){

int fd,fd1,fd2;

fd=socket(PF\_INET, SOCK\_STREAM,0);

fd1=open("test.dat",O\_CREAT|O\_WRONLY|O\_TRUNC);//open함수: 디스크립터 번>호 반환

fd2=socket(PF\_INET, SOCK\_DGRAM,0);

printf("file descriptor 1: %d \n",fd);

printf("file descriptor 2: %d \n",fd1);

printf("file descriptor 3: %d \n",fd2);

close(fd);

close(fd1);

close(fd2);

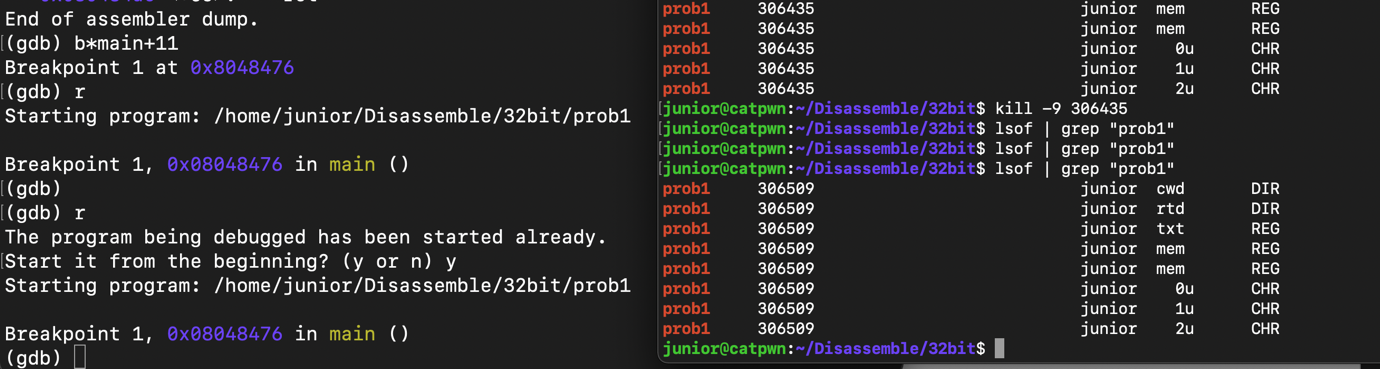
return 0;

}

//소켓: 데이터를 내보내거나 혹은 받기위한 창구

텍스트이(가) 표시된 사진

자동 생성된 설명



#calling convention 조사 및 정리

: 호출자(caller)와 피호출자(callee) 간의 함수의 인자를 전달하는 방식에 대한 규약을 정의한 것

1. \_\_cdecl 방식

:c/c++ 함수에서 사용

:호출자가 스택 정리

:오른쪽->왼쪽으로 인자 전달

:장점: 호출자가 인자 개수를 알고 있기 때문에 가변 인수 함수(함수의 인수와 개수 타입이 미리 정해져 있지 않음)를 만들 수 있다.

Ex.

테이블이(가) 표시된 사진

자동 생성된 설명

=

텍스트이(가) 표시된 사진

자동 생성된 설명

이처럼, cdecl 방식은 호출자가 피호출자의 스택 프레임을 정리한다는 것을 알 수 있다.

1. \_\_stdcall 방식

:win32 API 표준 규약

:호출자가 스택 정리

:오른쪽->왼쪽으로 인자 전달

:cdecl과 차이점: cdecl은 호출자가 스택 정리, stdcall은 피호출자가 직접정리

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

1. \_\_ fastcall 방식

:스택이 아닌 레지스터를 사용->속도 빠름

:레지스터 사용하기 때문에 따로 스택 정리 안함

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명