|  |
| --- |
| HW6. 구조체 배열  **분반: 061**  부산대학교 정보컴퓨터공학부  2015-24434  김영현  제출일: 2018-05-20 |

# 구현 내용에 대한 설명

주요 변수 구조체 Contact는 char형 이름배열과 char형 번호배열을 헤더파일 “phone.h”에서 선언하고, 이를 메인함수인 ‘PhoneBookMain.c’에서 구조체 배열 ‘PhoneBook[MAX]’로 정의한 후 다시 헤더파일에서 extern으로 전역변수로 설정하여 여러 소스파일에서 사용이 가능하다. 변수의 경우 extern을 사용해 전역변수로 설정하지 않는다면 다른 소스파일에서 사용할 수 없다. 반면에 함수는 전역 변수 설정이 디폴트이기 때문에 따로 extern 설정을 하지 않아도 메인소스파일에서 다른 소스파일의 함수를 전역함수처럼 사용이 가능하다. 만약 이러한 함수를 정의된 한 소스파일에서만 사용하기를 원한다면 static을 사용해 지역함수로 설정이 가능하다. “register.c”의 소스파일의 경우 password를 확인하는 함수 ‘in\_passwd()’를 해당 소스파일에서만 사용하기 위해 static설정을 하였고, 다른 소스파일에서 사용할 수 없다.

**-주요 함수 구현 방법**

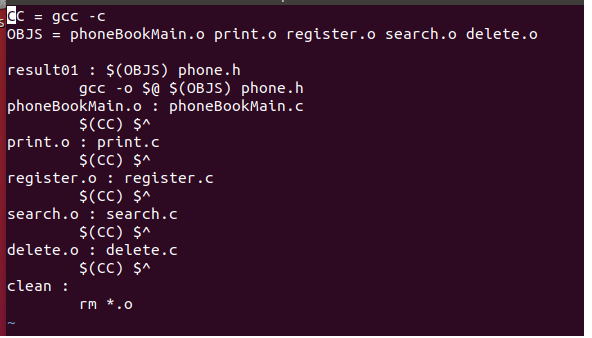
**Register() :** 연락처 리스트에 새로운 번호를 등록하는 함수이며, 이 함수는 연락처를 등록하기 전에 password를 확인하고나서 연락처를 구조체 배열에 등록한다. 전역변수인 size값을 배열의 index로 설정하고 scanf를 사용해 입력받은 후 다음 등록을 위해 size+1을 한 후 함수를 끝낸다.

**printAll() :** 연락처를 출력하는 함수로서 기존에 등록된 구조체 배열과 전역변수 size를 함수 내 지역변수 i로 입력 받는다. 현재 구조체 배열에 size-1 까지만 연락처 등록이 되어 있기 때문에 size-1 까지 출력한다. I의 경우 함수내의 지역변수이기에 이 함수가 종료되면 이 변수 또한 제거되기 때문에 후에 다시 출력한다 하여도 새로운 size 값을 받기 때문에 삭제된 배열까지 출력하는 상황을 방지할 수 있다.

**Search()** : 프로그램 사용자에게 이름을 입력받아 연락처 리스트에서 검색한 후 그에 해당하는 연락처를 출력하는 함수로서 <string.h>를 전처리하여 입력받은 이름을 strcmp(str1, str2)함수를 사용해 비교 후 그에 해당하는 index값으로 출력할 배열을 찾은 후 출력.

**Delete()** : 이 함수 또한 search()함수와 비슷한 형태를 가진다. Search()함수와 같이 이름을 입력받은 후 strcmp()함수로 index값을 찾는 과정까지는 동일하다. 여기서 phonebook[index+1]의 값을 strcpy(str1, str2)함수를 사용해 덮어 씌우고 index값을 (size-1)까지 1씩 증가시키며 반복문을 실행하고, 최종적으로 size값을 리스트의 마지막 index값인 (size-1)로 감소시키며 함수를 종료한다.

# 실행 방법 설명

 이번 과제는 Linux환경에서 vi를 사용해 소스코드 작성, gcc 컴파일러를 사용하였다. 컴파일의 경우 makefile을 작성하여 분할 컴파일을 진행하며, gcc -c 명령으로 각각의 소스파일의 오브젝트파일 ‘.o’를 생성 후 gcc -o 명령으로 5개의 오브젝트파일과 헤더파일을 포함시켜 실행파일 ‘PhoneBook’ 프로그램을 생성하였다. Makefile작성에 있어 suffix rule를 적용하여 한층 더 간결하게 작성하였고, 실행파일 생성 후 남게 되는 오브젝트 파일을 clean 명령어로 제거하기위해 추가하였다. 하나의 디렉토리에 소스파일과 헤더파일 그리고 makefile이 있으므로 terminal 에서 make 명령어와 make clean을 실행하였음.

# Github 화면

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명

# 논의 사항

위와 같은 프로그램을 작성하며 구조체 배열과 전역 함수, 전역 변수 등을 설정하며 프로그램에 간결성을 더하였다. 여기서 나아가 포인터를 추가하여 프로그램의 최적화를 진행하는데 있어 반복적인 오류로 인해 현재 제출한 버전에는 추가하지 못했으나 이후에 포인터 사용에 대해 오류점을 찾고 버전 업데이트를 하여 최적화를 진행할 예정이다. 현재 일반적인 변수 포인터 사용에는 명확한 개념을 가지며 적용하는데 어려움이 없으나, 이를 구조체 배열에 적용하여 정확하지 않은 방법으로 구조체 배열 포인터를 사용하는데 있어 지속적인 오류 발생으로 어려움이 있음.