|  |
| --- |
| HW6. 구조체 배열  **분반: 061**  부산대학교 정보컴퓨터공학부  201824465  문수빈  제출일: 2018-05-22 |

# 구현 내용에 대한 설명 (50점)

1. 주요 변수 설명: extern과 static 변수는 반드시 설명함

Phone.h에서 extern struct인 Contact 와 extern int형 변수 size를 선언하여 다른 파일에서도 사용할 수 있다. 최대 50개 정보를 저장할 수 있으므로 매크로 MAX를 설정하여 사용한다.

Register.c에서 실패한 횟수를 세는 변수 try를 static 변수로 설정하여 반복되는 함수 호출에도 다시 초기화 되지 않고 실패한 횟수를 유지하도록 한다.

1. 주요 자료 구조 설명

전화번호부 자료를 저장하는 구조체 Contact에 이름과 전화번호를 저장하는 char형 배열 멤버를 만든다. 구조체 배열을 사용하여 입력 받은 값을 저장한다.

비밀번호를 입력 받고 저장할 때, char형 배열 pw[10]을 사용한다.

검색할 이름을 입력 받고 저장할 때, char형 배열 search[10]을 사용한다.

삭제할 이름을 입력 받고 저장할 때, char형 배열 delname[10]을 사용한다.

1. 주요 함수 구현 방법 설명

registerPhoneData(); - while문을 실패한 횟수가 3번 이상일때까지 돌린다. Scanf를 통해 사용자가 비밀번호를 입력하고 내가 설정한 비밀번호와 일치하면 등록을 진행한다. 입력값과 설정값이 일치하는지 확인할 때는 strcmp 함수를 사용한다. 이름과 전화번호를 입력받고 size를 1만큼 증가시킨다.

void printAll(); - for문을 이용해 i=0부터 i=size까지 구조체 배열을 모두 출력한다.

void searchByName(); - 찾고 싶은 이름을 입력하고 입력 값과 일치하는 값을 가진 배열 첨자를 확인하기위해 for문을 이용해 i=0부터 i=size까지 찾는다. 중간에 일치하는 값이 나오면 break로 빠져나오고 없다면 일치하는 이름이 없다는 메시지를 출력한다.

void deleteByName(); - searchByName() 함수에서 구현한 것과 같이 입력 값과 일치하는 값을 찾은 후, 일치하는 값을 가진 배열에 그 다음 배열 값을 대입해 기존 값을 삭제하고 size도 1만큼 줄인다. 입력값과 일치하는 값이 있으면 break를 통해 빠져나오고 없다면 일치하는 이름이 없다는 메시지를 출력한다.

1. Makefile 설명

CFLAGS = -Wall -g

* Warning과 debugging을 포함하는 매크로

OBJS = register.o print.o search.o delete.o phoneBookMain.o

* Linking 할 파일들 매크로

all : phoneBookMain

* Makefile의 기본 target이 phoneBookMain

%.o : %.c

gcc -c -o $@ $(CFLAGS) $<

* \*.c 파일들을 \*.o로 컴파일한다.

phoneBookMain : $(OBJS)

gcc -o phoneBookMain $(OBJS)

* 매크로 OBJS에 담겨있는 파일들을 ‘phoneBookMain’에 linking한다.

clean:

rm phoneBookMain $(OBJS)

* 원하지 않는 파일을 삭제한다.

# 실행 방법 설명 (20점)

1. 사용한 운영체제 및 컴파일러의 종류

리눅스 / gcc

1. 컴파일 방법 및 실행 방법

분할컴파일로 makefile을 만들고 make 명령어로 컴파일한다.

./phoneBookMain 으로 실행한다.

1. 동작을 확인할 수 있는 실행 화면 캡처

![스크린샷이(가) 표시된 사진

자동 생성된 설명]()

# Github 화면 (20점)

1. cloning, adding, committing, push을 위한 github 명령들을 포함

![스크린샷이(가) 표시된 사진

자동 생성된 설명]()

![텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명]()

![텍스트이(가) 표시된 사진

자동 생성된 설명]()

1. 소스 코드와 makefile을 push한 후, 본인의 Github repository를 스크린 캡쳐하여 포함

![스크린샷이(가) 표시된 사진

자동 생성된 설명]()

# 논의 사항 (10점)

배열과 포인터에 관한 개념이 확실하지 않아서 조금 어려웠다. 처음에 입력 받는 비밀번호나 검색할 이름, 삭제할 이름 등을 char형 포인터를 사용해 문자열을 입력 받았다. 그래서 문자열을 비교할 때, 예를 들면 if (\*pw == \*password) 이렇게 비교했더니 같은 문자를 쳐도 else 문으로 빠져나갔다. 포인터는 그 문자열의 주소를 나타내므로 주소 값이 달라서 그렇다는 것을 알았다. char형 포인터 대신 char형 배열을 선언하고 strcmp 함수를 통해 비교했다.