|  |
| --- |
| HW6. 구조체 배열  **분반: 061**  부산대학교 사회과학대학 심리학과  201606151  김민승  제출일: 2018-05-14  1. 구현 내용에 대한 설명  (1) 주요 변수 설명  프로그램 구현에 있어서 전역변수 및 지역변수의 사용이 중요했었다. 주로 사용한 전역변수는 size, 구조체 배열(구조체 또한 변수의 집합이므로)이고, 나머지 변수들은 함수 블록 및 반복문 블록 내에서 사용하는 지역변수로 활용하였다.  특히, 여러 파일로 분할하여 컴파일하는 분할 컴파일 작업이라, extern 변수 선언의 활용이 중요했는데, main()함수를 갖는 phoneBookMain.c 소스 파일에서 정의한 전역변수 및 구조체를 다른 소스파일(모듈 파일)에서 불러와 사용하기 위해 각각 extern int size; extern struct Contact PhoneBook[];로 선언하여 사용하였다.  register.c 파일에서 블록 밖에 static char password[11] = “qwerty1234”; 코드를 입력했는데, 보안이 중요하고 다른 소스 파일에서 참조할 일이 없는 변수라 다른 소스 파일에서 참조하지 못하게끔 블록 밖에서 static 형태로 정의하였다.  (2) 주요 자료 구조 설명  전화번호부의 특성상 표(테이블)로 비유하자면 각 행(멤버)들이 데이터 형태가 다른 속성(필드)들을 가지고 있다. 이 과제의 경우에는 한 멤버는 두 개의 멤버 변수를 갖고있는 것으로 볼 수있다. 이를 구조체 배열을 활용하여, struct Contact{~~}로 구조체를 정의한 후, struct Contact PhoneBook[50]; 코드로 구조체를 선언하여 자료 구조를 구현한 것을 볼 수 있다.  (3) 주요 함수 구현 방법 설명  registerPhoneData()는, 반복문과 if 제어문을 활용하여 입력한 패스워드가 올바른 지 확인하고, 세 번 틀리면 함수를 종료하게끔 만든다. 이 때, String.h 헤더파일로 strcmp 함수를 사용하여 문자열을 비교한다. 세 번 내에 패스워드를 올바르게 입력하면 사용자로부터 User Name과 PhoneNumber를 입력받아 구조체에 저장한다. 이 때, 구조체 배열에서 size 번째의 인덱스에 저장하고, size를 1 증가시킨다. (size의 초기값은 0.) 여기서, size는 모든 소스 파일에서 extern을 활용하여 선언하여 전역변수로 사용함에 유의해야 한다.  printAll()은 단순 반복문을 활용하여 전역 변수인 size 값을 불러와 0번 인덱스부터 size-1번 째 인덱스 까지의 구조체 배열을 모두 출력하게끔 만든다.  searchByName()은 사용자로부터 문자열 하나(검색할 User Name)를 입력 받고, string.h로 사용가능한 strcmp 함수로 반복문을 활용하여 구조체 배열의 0번 인덱스부터 size-1 인덱스까지의 User Name을 비교하여 같으면 출력하고, 찾고자 하는 User Name이 없다면 그에 맞는 메시지를 출력하도록 한다.  deleteByName() 역시 문자열 복사 및 문자열 비교를 위해 string.h로 사용 가능한 strcmp와 strcpy를 활용한다. size 변수의 활용이 중요한데, 삭제하고자 하는 User Name 문자열을 사용자로부터 입력 받은 후, 이중 반복문을 사용한다. 바깥의 반복문은 첫 번째 인덱스부터 size-1 인덱스 까지 User Name을 비교하여, 같다면 삭제 메시지를 띄운 후에, 삭제하고자 하는 문자열과 같은 User Name을 가진 인덱스부터 size-1 번째 인덱스 까지 strcpy(PhoneBook[i].Name, PhoneBook[i+1].Name); strcpy(PhoneBook[i].PhoneNumber, PhoneBook[i+1].PhoneNumber); 코드로 앞의 인덱스가 가진 문자열을 뒤의 인덱스가 가진 문자열로 덮어씌운다. 그리고 size=size-1, index = -1로 초기화 하면, size가 1 줄고 검사하고자 하는 index는 처음부터 다시 시작하게 된다. 바깥의 반복문을 index가 size와 같아졌을 때 break;로 탈출시키면 삭제 함수가 구현이 가능하다. 그리고 삭제하고자 하는 문자열과 같은 User Name이 없다면 관련 메시지를 띄우고 함수를 종료시킨다.  (4) Makefile 설명  CC=gcc  CFLAGS=-Wall -g -c  INCLUDE=-I.  LIBS=-L.  OBJS = phoneBookMain.o register.o print.o search.o delete.o  all: PhoneBook  %.o: %.c  $(CC) $(INCLUDE) $(CFLAGS) $<  PhoneBook: $(OBJS)  $(CC) -o $@ $(OBJS) $(LIBS)  clean:  rm -f $(OBJS)  위와 같이 makefile을 작성하여 분할 컴파일하는데, 위의 네 줄은 매크로, 기본 target은 PhoneBook으로 설정한다. .o 목적파일은 .c 파일에 dependency를 두고, PhoneBook 실행 파일은 .o파일에 dependency를 둔다. clean: 항목은 컴파일한 후 불필요해진 목적 파일들을 삭제하게끔 만든다.  2. 실행 방법 설명  Linux (Ubuntu) 운영체제에서 gcc 컴파일러를 활용하였다.  위의 1.(4)에서 작성한 makefile을 활용하여 make 명령어로 자동 분할 컴파일하고, ./PhoneBook 명령어를 입력해 실행한다.  동작을 확인할 수 있는 실행 화면 캡처는 다음과 같다 :  스크린샷, 전화, 텍스트, 모니터이(가) 표시된 사진  자동 생성된 설명  3. Github 화면  Github 링크를 클릭한 후, repository에서 clone 버튼을 클릭하여 얻은 링크를 이용하여, Ubuntu에서 디렉토리를 하나 만든 후에 git clone https://github.com/pnusystemprog/osp-kor-hw6-kminssk.git 명령어를 입력하여 cloning하고,  텍스트 파일과 새로 생성한 코드들 및 수정한 기존의 cloning된 코드들을 git add \*.c git add \*.h git add \*.txt git add makefile 명령어를 이용하여 adding하고,  코딩을 완료한 후에 git -m commit “설명” 명령어를 입력하여 committing 후에 git push origin master 명령어를 입력하여 master 브랜치를 repository로 pushing했다.  Github 캡쳐화면은 다음과 같다 :  스크린샷이(가) 표시된 사진  자동 생성된 설명  4. 논의 사항  저번에 했던 과제4, 5에서 찾아보며 배웠던 헤더, 라이브러리, 선언과 정의의 차이, func prototype 등을 확실히 짚고 넘어 갔던 것이 도움이 크게 됐었던 것 같다. 만약, 저번 숙제에서 제대로 짚고 넘어가지 않았다면 이번 숙제부터 큰 난관에 봉착했을 것이라 생각하니, 스스로 찾아보며 개념을 파악하는 것이 중요하다는 것을 새삼 느꼈다.  숙제를 하면서 가장 어려운 난관이었던 것은, 한 소스 파일 안에서의 static의 사용법은 이해했으나, 다른 소스 파일을 참조하는 관점에서 static을 어떻게 사용할 것인지가 어려웠었다, 그러나 전역변수와 extern을 활용하면 굳이 static을 활용하지 않아도 될 것 같았고, 대신 static의 두 가지 활용 방법 중에서 블록 밖에서 선언하면 같은 소스파일안에서만 참조할 수 있는 static의 성질에서, register.c 파일에 있는 password 같이 중요한 변수의 경우 다른 소스파일에서 참조하지 못하게끔 하는 역할도 나중에 실제로 활용할 때 보안 차원에서 중요하겠구나 하는 것을 깨달았다.  그리고 size와 같은 전역변수를 어떻게 활용할 것인지가 생각보다 복잡했고, 포인터의 개념을 활용하면 보다 쉽게 할 수 있지 않을까 하는 생각을 하게 되었다.  문자열 비교 및 복사를 함에 있어서 단순히 = 연산자를 활용하고자 했으나 c언어에서는 문자열을 다루는 것에 있어서 확실히 부족함이 많음을 알았고, 대신 도움을 주는 여러가지 문자열 관련 함수가 있는 것을 알게 되었다. C에서 보다 문자열을 쉽게 다루기 위해서 어떤 방법이 있을까 하고 고민해보게 되었다. |