[Database system Assignment 1]

연락처 관리 시스템 만들기

컴퓨터소프트웨어

2015005141 최연수

1. 개발환경

Apple clang version 11.0.0 (clang-1100.0.33.8)

Terminal, gcc 4.2.1

2. 사용언어

C언어

3. 코드설명

3.1 아이디어

'Linked-list' 자료구조를 이용하여 검색 삽입 삭제 수정 을 용이하게 한다.

3.2 검색

```
list Find(list person, char name[]){        //find the name in the list person

    list tmp, tmp2;
    char finally_find[100];
    int check=0;
    tmp = person;

    while(tmp != NULL){

        if(check_same(name, tmp->name)){        //print all name matched with input_name
            check++;                            //for checking whether put whole name which is the only or not.
            printf("%2d.%s  ",check,tmp->name);
            printf("%s\n",tmp->phone);
            if(check==1){       //in case put whole name which is the only. to store it.
                tmp2 = tmp;
            }
        }
        tmp = tmp->next;
    }
```

Find 함수에서 현재 저장되어있는 list와 검색 할 name을 인자로 받는다.

Check_same 함수(본문 3.6)를 통해 조건을 만족하면, 인수로 받은 name을 포함하고 있는 이름들을 모두 출력한다.

유일한 이름을 입력 할 경우에 대비해 처음 값을 따로 저장해 둔다.

```
    if(check >1){       //it means there are many options
        printf("*Please collect Who you want to change (enter the name) : ");
        scanf("%s",finally_find);
        tmp = person;
        check = 0;
        while(tmp != NULL && strcmp(finally_find, tmp->name)!=0){
            check++;
            tmp = tmp->next;
        }

        if(tmp == NULL){
            printf("error : %s is not in the list\n",finally_find);
            printf("program is ended\n");
            exit(-1);
        }
    }


    else if(check == 1){
        return tmp2;
    }


    return tmp;

}
```

일치하는 항목이 여러 개 있을 경우(check > 1), 정확한 이름을 입력 받아 그를 반환한다.

항목에 존재하지 않는 이름을 입력할 경우 error message를 출력하고 프로그램을 종료한다.

일치하는 항목이 유일하게 있을 경우(check == 1), 일전에 저장 해놓았던 값을 반환한다.

3.3 삽입

```
    printf("*%s is not in the list. please add\n",input_name);
    printf("phone number : ");
    scanf("%s", input_p_num);

    if(pnum_check(input_p_num)){
        insert(person, input_name, input_p_num);
    }

    else{
        printf("*please put 11 numbers start with 010\n");
        break;
    }
}
```

Main 함수에서 pnum_check함수(본문 3.7)의 조건을 만족하면 삽입을 시행한다.

```
void insert(list L, char name[], char p_num[]){

    list tmp;
    tmp = malloc(sizeof(struct List));
    if(tmp == NULL){
        printf("insert error\n");
        exit(-1);
    }
    strcpy(tmp->name, name);
    strcpy(tmp->phone, p_num);

    tmp -> next = L -> next;
    L -> next = tmp;
}
```

Insert함수에서, 저장된 리스트와 새롭게 저장 할 이름과 번호를 인자로 받는다.

새로운 list tmp에 공간을 할당하고, 인자로 받은 값들을 입력해준다.

이후 인자로 받은 리스트에 연결해준다.

3.4 삭제

```
case 2:
    printf("*who do you want to delete? : ");
    scanf("%s", input_name);
    Delete(person, input_name, 1);
    printf("------delete complete------\n");
    break;
```

main 함수에서 삭제 할 이름을 입력받아 삭제를 시행한다

```
list Find_pre(list person, char name[], int d_check){       //using in delete()
    list tmp,tmp2;
    char finally_find[100];
    int check=0;
    tmp = person;

    while(tmp->next != NULL){

        if(check_same(name, tmp->next->name)){
            check++;
            if(d_check != 2){
                printf("%2d.%s  ",check,tmp->next->name);
                printf("%s\n",tmp->next->phone);
            }
            if(check==1){
                tmp2 = tmp;

            }
        }

        tmp = tmp->next;

    }
```

Find_pre 함수에서 현재 리스트와 삭제할 이름을 인자로 받는다.

추가적으로, 출력을 용이하게 하기 위해 int형 변수 d_check을 인자로 받는다.(본문 3.5)

리스트를 돌며, 입력 받은 이름을 포함한 항목 이전 노드를 모두 찾는다. (check_same())

이때, 수정과 삭제를 구분하기위해 d-check이 2가 아닐때만 항목들을 출력한다.(본문 3.5)

유일한 이름을 입력 할 경우에 대비해 처음 값을 따로 저장해둔다.

```
    if(check >1){
        printf("Please collect Who you want to delete (enter the name) : ");
        scanf("%s",finally_find);
        tmp = person;
        check = 0;
        while(tmp->next != NULL && strcmp(finally_find, tmp->next->name)!=0){
            tmp = tmp->next;
            check++;
        }
        if(check == 0){
            tmp -> next = NULL;
        }
    }

    else if(check == 1){
           return tmp2;
    }

    else if(check == 0)
        tmp->next = NULL;

    return tmp;

}
```

일치하는 항목이 여러 개 있을 경우, 정확한 이름을 입력 받아 그를 반환한다.(check > 1)

항목에 존재하지 않는 이름을 입력할 경우 tmp->next에 NULL값을 입력하여 error message를 출력하게 한다..(check == 0)

일치하는 항목이 유일하게 있을 경우, 일전에 저장 해놓았던 값을 반환한다(check == 1)

```
void Delete(list person, char name[], int d_check){
    list tmp,p;
    tmp = Find_pre(person, name, d_check);
    if(tmp->name != NULL){
        p = tmp->next;
        tmp -> next = p -> next;
        free(p);
    }
    else
        printf("delete fail. that's not in the list\n");

}
```

Delete 함수에서 현재 리스트와 삭제할 이름을 인자로 받는다.

추가적으로, 출력을 용이하게 하기 위해 int형 변수 d_check을 인자로 받는다.(본문 3.5)

Find_pre 함수를 사용하여, 찾고자 하는 노드 이전의 노드를 찾는다.

삭제하고자 하는 노드를 free()해주면서 함수를 완성한다.

일치하는 항목이 없으면 error 메세지를 출력한다.

3.5 수정

```
switch (button){
    case 1:
        printf("*Who do you want to find : ");
        scanf("%s",input_name);


        tmp_list = Find(person, input_name);

        if(tmp_list->name){

            printf("----------------------\n");
            printf("What do you want to change?\n");
            printf("1.name\n");
            printf("2.phone number\n");
            printf("3.nothing\n");
            printf("----------------------\n");

            scanf("%d",&second_button);

            if(second_button == 1){
                printf("*please update %s's name : ", tmp_list->name);      //update the name

                char input_name[] = {'\0',};
                scanf("%s",input_name);
                strcpy(remain_p_num , tmp_list->phone);
                Delete(person, tmp_list->name, 2);
                insert(person, input_name, remain_p_num);
            }

            else if(second_button == 2){
                printf("*please update %s's phone number : ", tmp_list->name);      //update the phone_number

                char input_p_num[] = {'\0',};
                scanf("%s",input_p_num);

                if(pnum_check(input_p_num)){
                    strcpy(remain_name, tmp_list->name);
                    Delete(person, tmp_list->name, 2);
                    insert(person, remain_name, input_p_num);

                }
```

수정을 원하는 항목을 삭제하고, 수정한 항목을 새롭게 삽입한다.

항목 삭제 시, find_pre함수에서의 출력을 방지하기위해 d_check 값으로 2를 넘겨준다.

번호를 수정하고 싶을 땐, pnum_check 함수를 이용해서 정규화 된 번호만 입력 받게 한다.(본문3.7)

3.6 부분검색

```
int check_same(char input_name[], char name_in_file[]){   // using in find() & delete()
                                                          // to search using just partial name
    int length = strlen(input_name);
    int count=0;

    for(int i=0 ; i<length ; i++){
        if(input_name[i] == name_in_file[i]){
            count++;
        }
    }

    if(count == length)
        return 1;
    else
        return 0;
}
```

Check_same함수에서, 검색을 위해 입력받은 값과, 파일에 저장 되어있는 이름을 인자로 받는다.

입력받은 값의 길이만큼 for문을 돌며 파일에 저장된 값과 같은지 확인한다.

같다면(입력받은 값의 길이만큼 for문이 진행됐다면) 참을 표시하기위해 1을 반환한다.

3.7 번호 정규화

```
    int pnum_check(char p_num[]){   //to put 11 numbers start with '010'

        int number;
        char tmp;

        for(int i=0 ; i<3 ; i++){

            tmp = p_num[i];
            number = tmp - '0';

            if(number != i % 2){      // 0%2 ==0, 1%2 == 1, 2%2 == 0
                return 0;
            }
        }

        if(strlen(p_num) != 11){
            return 0;
        }

        return 1;
    }
```

pnum_check 함수에서, 수정 혹은 삽입을 위해 입력 받은 phone number을 인자로 받는다.

Char 타입과 int 타입을 비교하기위해 int형 변수 number 에 번호문자열(tmp) – '0' 을 입력해준다.

Phone number의 처음 3자리가 '010' 이 아니게 되는 순간 0을 반환한다.

Strlen 함수를 이용하여 길이가 11이 넘어가면 0을 반환한다.

두 항목 모두 해당하지 않는다면 1을 반환한다.

3.8 기타구현

3.8.1 구조체 생성

```
struct List{
    char name[61];
    char phone[12];
    position next;
};
```

Name 배열의 크기를 61로 잡아, 최대 20글자의 이름을 받도록 제한한다.

Phone 배열의 크기를 12로 잡아, 최대 11자리의 번호를 받도록 제한한다.

3.8.2 header 생성

```
list create(list L){
    L = (list)malloc(sizeof(struct List));
    strcpy(L -> name,"name");
    strcpy(L -> phone,"phone");
    L->next = NULL;

    return L;
}
```

구조체 크기만큼 공간할당을 해준 뒤, 이름을 name. 휴대폰 번호를 phone.

Next값을 NULL로 초기화하여 linked-list의 header를 생성해준다.

3.8.3 파일 입출력

```
char *file_name  = "2015005141_최연수.csv";
char str[] = ",\r\n";    // use \r in .csv file but I use \n.

file_exist = access(file_name, F_OK);

if(file_exist == 0)         //file exist
    fp = fopen( "2015005141_최연수.csv","r+" );
else
    fp = fopen( "contact.csv","r+" );


if ( fp != NULL){

    while ( fgets(str_tmp, 1024, fp) != NULL ){

        char name[61] = {'\0'};
        char p_number[12] = {'\0'};

        tmp = strtok(str_tmp, str);
        if(strcmp(tmp, "name")==0) continue;

        strcpy(name , tmp);

        tmp = strtok(NULL, str);
        strcpy(p_number , tmp);

        insert(person, name, p_number);
    }
}

else{
    printf("Open error\n");
    return 0;
}

fclose(fp);
```

Str[]에 , ₩n ₩r 을 저장하여, 해당 문자가 나오면 문자열을 끊는다..

"2015005141_최연수.csv"파일이 존재한다면 읽기 형식으로 해당 파일을 열어준다.(fopen ,r)

존재하지 않는다면 "contact.csv" 파일을 열어준다.(fopen, r)

파일이 끝날 때 까지 문장을 읽어들이며 리스트에 저장(insert) 후 파일을 닫아준다.

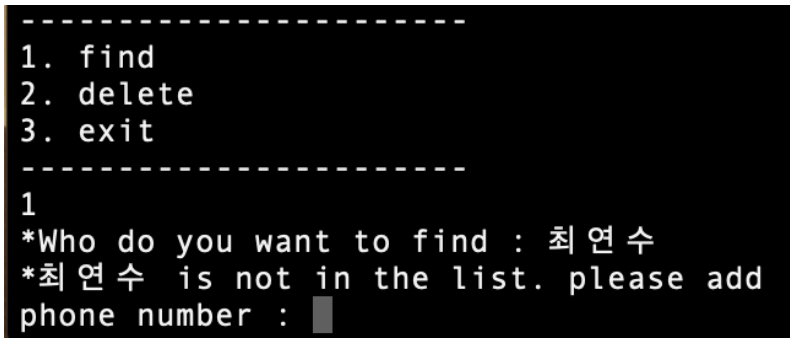
```
new_fp = fopen("2015005141_최연수.csv","w");

while(person != NULL){
    fputs(person->name, new_fp);
    fputs(",",new_fp);
    fputs(person->phone,new_fp);
    if(person->next !=NULL)
        fputs("\n",new_fp);
    person = person->next;
}

fclose( new_fp );
```

작업이 완료됐다면 쓰기 형식으로 "2015005141_최연수.csv"파일을 열어준다.

저장되어있는 리스트를 형식에 맞게 출력해준다(fputs).

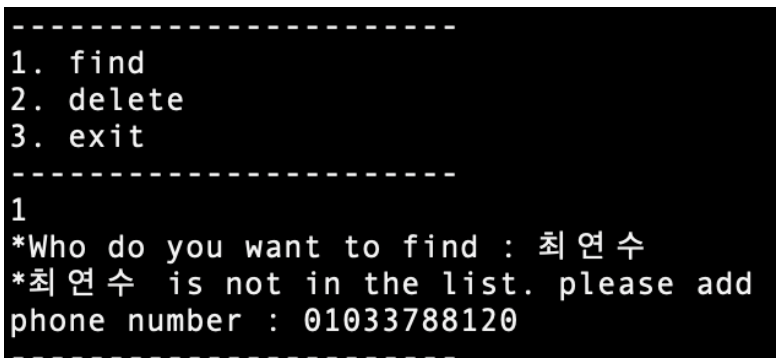
4 시나리오

4.1 실행 후 검색



검색한 이름이 리스트에 없어서, 새롭게 추가하라는 메세지가 출력된다.

4.2 삽입



전화번호 입력을 통해 추가해준다.

4.3 재 시작 후 검색

```
-----------------------
1. find
2. delete
3. exit
-----------------------
3
yeonsooui-MacBookPro:~ yeonsoochoi$ ./a.out
-----------------------
1. find
2. delete
3. exit
-----------------------
1
*Who do you want to find : 최연수
 1.최연수   01033788120
```

프로그램 재 시작 후 검색을 해보면, 연락처가 추가된 것을 볼 수 있다.

4.4 삭제

```
*Who do you want to find : 최연수
 1.최연수   01033788120
-----------------------
What do you want to change?
1.name
2.phone number
3.nothing
-----------------------
3
-----------------------
1. find
2. delete
3. exit
-----------------------
2
*who do you want to delete? : 최연수
 1.최연수   01033788120
------delete complete------
```

4.5 프로그램 재 시작 후 검색

```
yeonsooui-MacBookPro:~ yeonsoochoi$ ./a.out
-----------------------
1. find
2. delete
3. exit
-----------------------
1
*Who do you want to find : 최연수
*최연수  is not in the list. please add
phone number :
```

삭제가 성공적으로 이루어진 것을 볼 수 있다.

5 추가 시나리오

5.1 이름 수정

```
-----------------------
1. find
2. delete
3. exit
-----------------------
1
*Who do you want to find : 최연수
 1.최연수   01033788120
-----------------------
What do you want to change?
1.name
2.phone number
3.nothing
-----------------------
1
*please update 최연수 's name : 가나다라마바사
-----------------------
1. find
2. delete
3. exit
-----------------------
1
*Who do you want to find : 가나다라마바사
 1.가나다라마바사   01033788120
```

5.2 부분검색

```
----------------------
1. find
2. delete
3. exit
----------------------
1
*Who do you want to find : 최
 1.최연수   01033788120
 2.최병복   01034576069
 3.최용돈   01075074228
 4.최순복   01045447385
 5.최지안   01023878086
 6.최운호   01053695582
 7.최성하   01042366525
 8.최숙남   01090463174
 9.최준한   01029378046
10.최일승   01090376687
11.최대섭   01084573746
12.최춘영   01048767314
13.최근섭   01047088451
14.최선례   01062200300
15.최홍균   01058526650
```

```
1873.최영종   01029937524
1874.최복녀   01085774136
1875.최완하   01080414897
1876.최문락   01025483518
Please collect Who you want to change (enter the name) :
```

성씨 검색으로, 성씨가 같은 인물들이 모두 출력되는 것을 볼 수 있다.

```
----------------------
1. find
2. delete
3. exit
----------------------
2
*who do you want to delete? : 최연
 1.최연수   01033788120
 2.최연경   01059968179
 3.최연중   01044470989
 4.최연관   01057068271
 5.최연승   01057978284
 6.최연우   01053078954
 7.최연출   01063194491
 8.최연오   01025590948
 9.최연만   01024953162
10.최연심   01046666515
11.최연호   01060156597
12.최연철   01084379415
13.최연근   01064934159
14.최연주   01087060045
15.최연희   01049657143
16.최연대   01061444216
17.최연규   01039602725
18.최연암   01060939213
19.최연정   01081138957
20.최연묵   01068274910
21.최연성   01022070568
22.최연섭   01064586577
Please collect Who you want to delete (enter the name) : 최연수
```

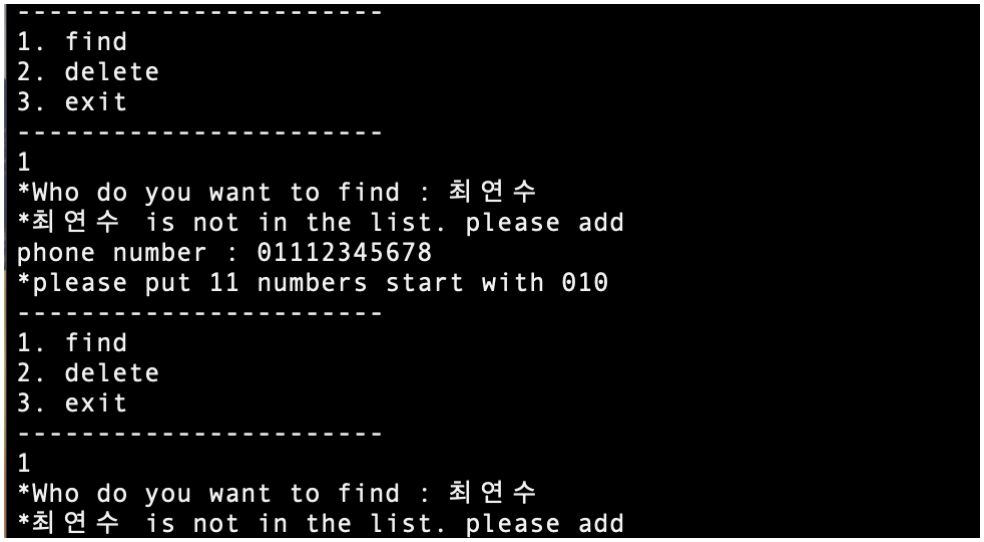
삭제 부분에서도 부분 검색이 가능하다.

```
----------------------
1. find
2. delete
3. exit
----------------------
1
*Who do you want to find : 최연
 1.최연경   01059968179
 2.최연중   01044470989
 3.최연관   01057068271
 4.최연승   01057978284
 5.최연우   01053078954
 6.최연출   01063194491
 7.최연오   01025590948
 8.최연만   01024953162
 9.최연심   01046666515
10.최연호   01060156597
11.최연철   01084379415
12.최연근   01064934159
13.최연주   01087060045
14.최연희   01049657143
15.최연대   01061444216
16.최연규   01039602725
17.최연암   01060939213
18.최연정   01081138957
19.최연묵   01068274910
20.최연성   01022070568
21.최연섭   01064586577
*Please collect Who you want to change (enter the name) : 가나다
error : 가나다  is not in the list
program is ended
```

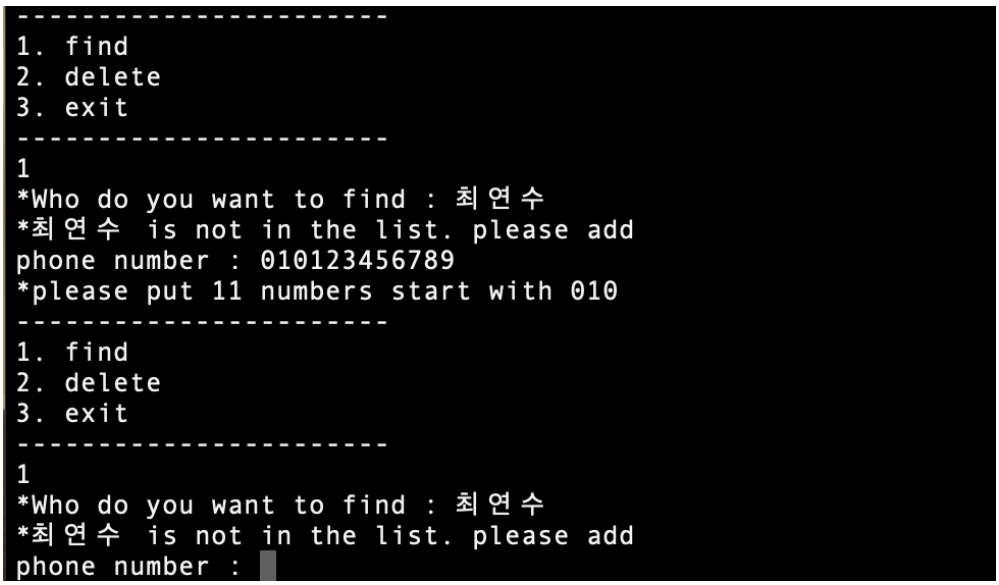
```
----------------------
1. find
2. delete
3. exit
----------------------
2
*who do you want to delete? : 최연
 1.최연경   01059968179
 2.최연중   01044470989
 3.최연관   01057068271
 4.최연승   01057978284
 5.최연우   01053078954
 6.최연출   01063194491
 7.최연오   01025590948
 8.최연만   01024953162
 9.최연심   01046666515
10.최연호   01060156597
11.최연철   01084379415
12.최연근   01064934159
13.최연주   01087060045
14.최연희   01049657143
15.최연대   01061444216
16.최연규   01039602725
17.최연암   01060939213
18.최연정   01081138957
19.최연묵   01068274910
20.최연성   01022070568
21.최연섭   01064586577
22.최연수   01033788120
Please collect Who you want to delete (enter the name) : 가나다
delete fail. that's not in the list
```

부분 검색 시 목록에 없는 이름을 입력하면 에러 메세지가 출력된다.

5.3 번호 정규화



'010'으로 시작하지 않는 전화번호를 입력했을 때, 에러메세지가 출력되고, 수정(삽입)이 되지 않는다.



전화번호가 11자리가 아니면, 에러메세지가 출력되고 수정(삽입)이 되지 않는다.