문제해결기법(13967005)

202135592 한웅재

소프트웨어

제출일: 2021. 11. 7

Q1. Lab-S (p.10) : terminal screenshot

#define \_CRT\_SECURE\_NO\_WARNINGS// or scanf\_s

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <ctype.h>

#include <stdbool.h>

struct VOTE {

char region[20];

int age;

char candidate\_voted[20];

};

struct VOTE vote[2000];

void copy\_element(struct VOTE src[], struct VOTE dest[]) {

strcpy(dest->region, src->region);

dest->age = src->age;

strcpy(dest->candidate\_voted, src->candidate\_voted);

}

bool read\_file(const char\* fname) {

FILE\* pFile;

pFile = fopen(fname, "r");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

struct VOTE person;

int i = 0;

while (fscanf(pFile, "%s %d %s", person.region, &person.age, person.candidate\_voted) == 3) {

copy\_element(&person, &vote[i]);

i++;

}

fclose(pFile);

return true;

}

void Compute\_vote(struct VOTE\* vote,

float\* Washington,

float\* Lincoln,

float\* Roosevelt,

int\* num\_of\_voter)

{

float Washington\_voted = 0;

float Lincoln\_voted = 0;

float Roosevelt\_voted = 0;

for (int i = 0; i < 2000; i++) {

if (strcmp(vote[i].candidate\_voted, "Washington") == 0) {

Washington\_voted++;

(\*num\_of\_voter)++;

}

else if (strcmp(vote[i].candidate\_voted, "Lincoln") == 0) {

Lincoln\_voted++;

(\*num\_of\_voter)++;

}

else if (strcmp(vote[i].candidate\_voted, "Roosevelt") == 0) {

Roosevelt\_voted++;

(\*num\_of\_voter)++;

}

}

\*Washington = (Washington\_voted / \*num\_of\_voter) \* (float)100.0;

\*Lincoln = (Lincoln\_voted / \*num\_of\_voter) \* (float)100.0;

\*Roosevelt = (Roosevelt\_voted / \*num\_of\_voter) \* (float)100.0;

}

void Compute\_third\_row\_vote(struct VOTE\* vote,

float\* Washington,

float\* Lincoln,

float\* Roosevelt,

int\* num\_of\_voter) {

\*num\_of\_voter = 0;

float Washington\_voted = 0;

float Lincoln\_voted = 0;

float Roosevelt\_voted = 0;

for (int i = 0; i < 2000; i++) {

if (i % 3 == 0) {

if (strcmp(vote[i].candidate\_voted, "Washington") == 0) {

Washington\_voted++;

(\*num\_of\_voter)++;

}

else if (strcmp(vote[i].candidate\_voted, "Lincoln") == 0) {

Lincoln\_voted++;

(\*num\_of\_voter)++;

}

else if (strcmp(vote[i].candidate\_voted, "Roosevelt") == 0) {

Roosevelt\_voted++;

(\*num\_of\_voter)++;

}

}

}

\*Washington = (Washington\_voted / \*num\_of\_voter) \* (float)100.0;

\*Lincoln = (Lincoln\_voted / \*num\_of\_voter) \* (float)100.0;

\*Roosevelt = (Roosevelt\_voted / \*num\_of\_voter) \* (float)100.0;

}

void Compute\_tenth\_row\_vote(struct VOTE\* vote,

float\* Washington,

float\* Lincoln,

float\* Roosevelt,

int\* num\_of\_voter) {

\*num\_of\_voter = 0;

float Washington\_voted = 0;

float Lincoln\_voted = 0;

float Roosevelt\_voted = 0;

for (int i = 0; i < 2000; i++) {

if (i % 10 == 0) {

if (strcmp(vote[i].candidate\_voted, "Washington") == 0) {

Washington\_voted++;

(\*num\_of\_voter)++;

}

else if (strcmp(vote[i].candidate\_voted, "Lincoln") == 0) {

Lincoln\_voted++;

(\*num\_of\_voter)++;

}

else if (strcmp(vote[i].candidate\_voted, "Roosevelt") == 0) {

Roosevelt\_voted++;

(\*num\_of\_voter)++;

}

}

}

\*Washington = (Washington\_voted / \*num\_of\_voter) \* (float)100.0;

\*Lincoln = (Lincoln\_voted / \*num\_of\_voter) \* (float)100.0;

\*Roosevelt = (Roosevelt\_voted / \*num\_of\_voter) \* (float)100.0;

}

int main() {

char fname[50] = "vote.txt";

int num\_of\_voters = 0;

float Washington;

float Lincoln;

float Roosevelt;

read\_file(fname);

Compute\_vote(vote, &Washington, &Lincoln, &Roosevelt, &num\_of\_voters);

printf("Vote Result: Washington : %.1f%% Lincoln : %.1f%% Roosevelt : %.1f%% , number of voters : %d\n", Washington, Lincoln, Roosevelt, num\_of\_voters);

Compute\_third\_row\_vote(vote, &Washington, &Lincoln, &Roosevelt, &num\_of\_voters);

printf("Vote Result(EVERY THIRD ROW): Washington : %.1f%% Lincoln : %.1f%% Roosevelt : %.1f%% , number of voters : %d\n", Washington, Lincoln, Roosevelt, num\_of\_voters);

Compute\_tenth\_row\_vote(vote, &Washington, &Lincoln, &Roosevelt, &num\_of\_voters);

printf("Vote Result(EVERY TENTH ROW): Washington : %.1f%% Lincoln : %.1f%% Roosevelt : %.1f%% , number of voters : %d\n", Washington, Lincoln, Roosevelt, num\_of\_voters);

return 0;

}

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

Q2. Lab-H (p.13) : output.txt screenshot

#define \_CRT\_SECURE\_NO\_WARNINGS// or scanf\_s

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <ctype.h>

#include <stdbool.h>

struct PERSONAL {

char name[20];

int age;

char hobby[20];

};

struct PERSONAL personal[20];

struct PERSONAL decomposed\_data[20];

void copy\_element(struct PERSONAL src[], struct PERSONAL dest[]) {

strcpy(dest->name, src->name);

dest->age = src->age;

strcpy(dest->hobby, src->hobby);

}

void group\_by\_age(struct PERSONAL\* original, struct PERSONAL\* age) {

int index = 0;

for (int k = 1; k < 5; k++) {

for (int i = 0; i < 20; i++) {

int p\_age = (original[i].age / 10) \* 10;

if (p\_age == k \* 10) {

copy\_element(&original[i], &age[index]);

index++;

}

}

}

}

bool write\_file(const char\* fname, struct PERSONAL edit[]) {

FILE\* pFile;

pFile = fopen(fname, "w");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

for (int i = 0; i < 20; i++) {

if (i == 0) {

fprintf(pFile, "Age from %d to %d\n--------------------------\n", (edit[i].age / 10) \* 10, (edit[i].age / 10) \* 10 + 9);

}

if (i >= 1 && i <= 19) {

if (edit[i].age / 10 - edit[i - 1].age / 10) {

fprintf(pFile, "\nAge from %d to %d\n--------------------------\n", (edit[i].age / 10) \* 10, (edit[i].age / 10) \* 10 + 9);

}

}

fprintf(pFile, "%8s %d %8s\n",

edit[i].name,

edit[i].age,

edit[i].hobby);

}

fclose(pFile);

return true;

}

bool read\_file(const char\* fname) {

FILE\* pFile;

pFile = fopen(fname, "r");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

struct PERSONAL person;

int i = 0;

while (fscanf(pFile, "%s %d %s", person.name, &person.age, person.hobby) == 3) {

copy\_element(&person, &personal[i]);

i++;

}

fclose(pFile);

return true;

}

int main() {

char fname[50] = "personal.txt";

char output\_name[30] = "output.txt";

read\_file(fname);

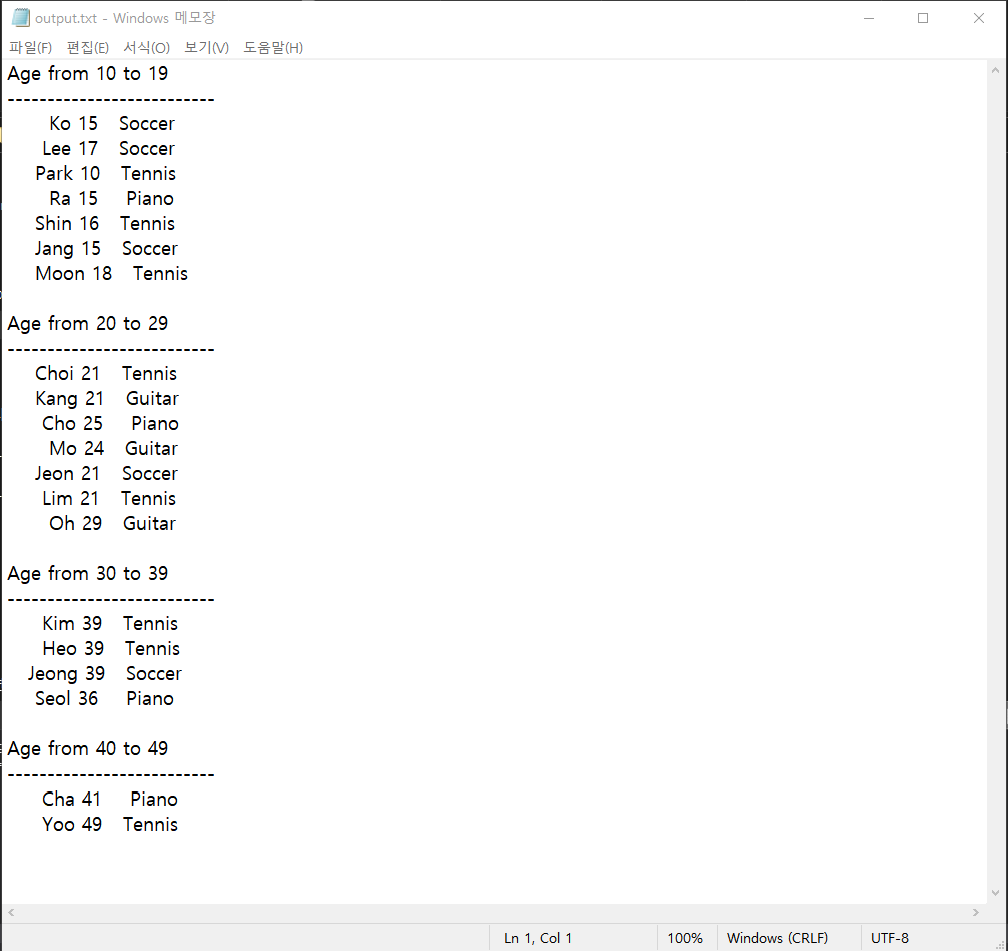
group\_by\_age(personal, decomposed\_data);

write\_file(output\_name, decomposed\_data);

return 0;

}

텍스트이(가) 표시된 사진

자동 생성된 설명

Q3. Lab-V (p.15) : age.txt & hobby.txt screenshots

#define \_CRT\_SECURE\_NO\_WARNINGS// or scanf\_s

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <ctype.h>

#include <stdbool.h>

struct PERSONAL {

char name[20];

int age;

char hobby[20];

};

struct NAME\_AGE {

char name[20];

int age;

};

struct NAME\_HOBBY {

char name[20];

char hobby[20];

};

struct PERSONAL personal[20];

struct NAME\_AGE Age[20];

struct NAME\_HOBBY Hobby[20];

void copy\_element(struct PERSONAL src[], struct PERSONAL dest[]) {

strcpy(dest->name, src->name);

dest->age = src->age;

strcpy(dest->hobby, src->hobby);

}

void copy\_age\_element(struct PERSONAL src[], struct NAME\_AGE dest[]) {

strcpy(dest->name, src->name);

dest->age = src->age;

}

void copy\_hobby\_element(struct PERSONAL src[], struct NAME\_HOBBY dest[]) {

strcpy(dest->name, src->name);

strcpy(dest->hobby, src->hobby);

}

bool write\_file(const char\* fname, struct PERSONAL edit[]) {

FILE\* pFile;

pFile = fopen(fname, "w");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

for (int i = 0; i < 20; i++) {

fprintf(pFile, "%8s %d %8s\n",

edit[i].name,

edit[i].age,

edit[i].hobby);

}

fclose(pFile);

return true;

}

bool write\_file\_for\_age(const char\* fname, struct NAME\_AGE edit[]) {

FILE\* pFile;

pFile = fopen(fname, "w");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

for (int i = 0; i < 20; i++) {

fprintf(pFile, "%8s %d\n",

edit[i].name,

edit[i].age);

}

fclose(pFile);

return true;

}

bool write\_file\_for\_hobby(const char\* fname, struct NAME\_HOBBY edit[]) {

FILE\* pFile;

pFile = fopen(fname, "w");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

for (int i = 0; i < 20; i++) {

fprintf(pFile, "%8s %8s\n",

edit[i].name,

edit[i].hobby);

}

fclose(pFile);

return true;

}

bool read\_file(const char\* fname) {

FILE\* pFile;

pFile = fopen(fname, "r");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

struct PERSONAL person;

int i = 0;

while (fscanf(pFile, "%s %d %s", person.name, &person.age, person.hobby) == 3) {

copy\_element(&person, &personal[i]);

i++;

}

fclose(pFile);

return true;

}

bool read\_file\_for\_age(const char\* fname, struct NAME\_AGE\* Age) {

FILE\* pFile;

pFile = fopen(fname, "r");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

struct PERSONAL person;

int i = 0;

while (fscanf(pFile, "%s %d %s", person.name, &person.age, person.hobby) == 3) {

copy\_age\_element(&person, &Age[i]);

i++;

}

fclose(pFile);

return true;

}

bool read\_file\_for\_hobby(const char\* fname, struct NAME\_HOBBY\* Hobby) {

FILE\* pFile;

pFile = fopen(fname, "r");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

struct PERSONAL person;

int i = 0;

while (fscanf(pFile, "%s %d %s", person.name, &person.age, person.hobby) == 3) {

copy\_hobby\_element(&person, &Hobby[i]);

i++;

}

fclose(pFile);

return true;

}

int main() {

char fname[50] = "personal.txt";

char fage[30] = "age.txt";

char fhobby[30] = "hobby.txt";

read\_file(fname);

read\_file\_for\_age(fname,Age);

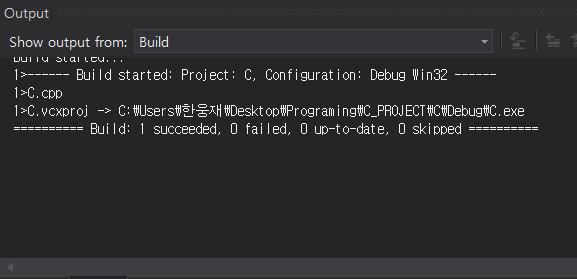
read\_file\_for\_hobby(fname, Hobby);

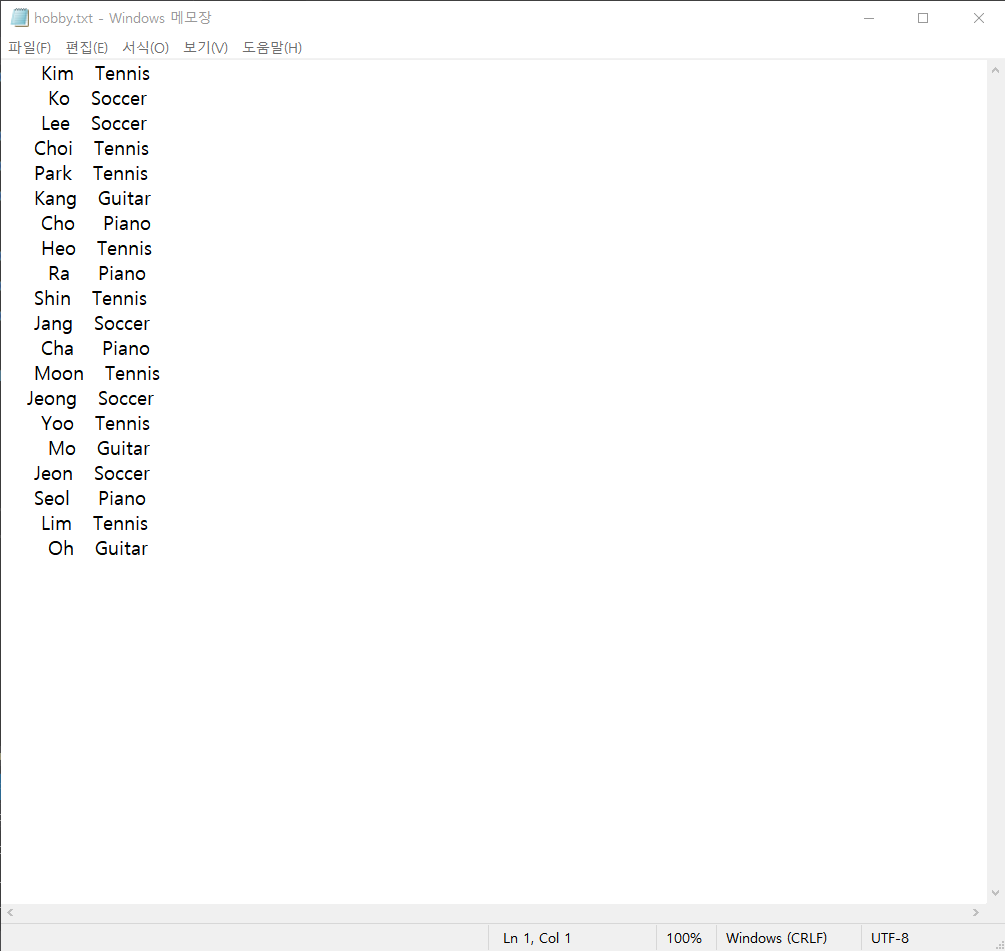
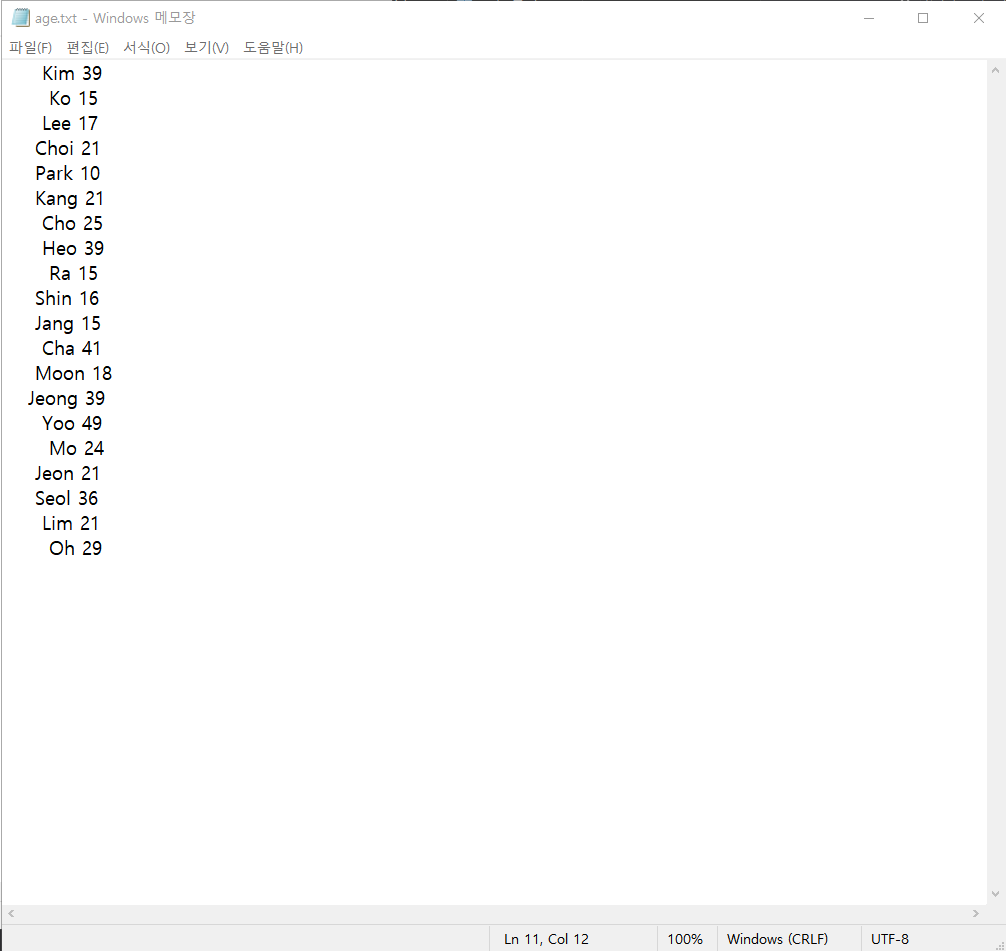
write\_file\_for\_age(fage,Age);

write\_file\_for\_hobby(fhobby,Hobby);

return 0;

}





Q4. Lab-VS (p.19,20) : salary\_v2.txt, salary\_v3.txt screenshots + terminal screenshot

#define \_CRT\_SECURE\_NO\_WARNINGS// or scanf\_s

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <ctype.h>

#include <stdbool.h>

struct PERSONAL {

char name[20];

int age;

double salary;

};

struct PERSONAL personal\_1[20];

struct PERSONAL personal\_2[20];

struct PERSONAL personal\_3[20];

struct PERSONAL temp[20];

void copy\_element(struct PERSONAL src[], struct PERSONAL dest[]) {

strcpy(dest->name, src->name);

dest->age = src->age;

dest->salary = src->salary;

}

bool write\_file(const char\* fname, struct PERSONAL edit[]) {

FILE\* pFile;

pFile = fopen(fname, "w");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

for (int i = 0; i < 20; i++) {

fprintf(pFile, "%8s %d %.1lf\n",

edit[i].name,

edit[i].age,

edit[i].salary);

}

fclose(pFile);

return true;

}

bool read\_file(const char\* fname,struct PERSONAL\* data) {

FILE\* pFile;

pFile = fopen(fname, "r");

if (pFile == NULL) {

printf("cannot open the file!\n");

return false;

}

struct PERSONAL person;

int i = 0;

while (fscanf(pFile, "%s %d %lf", person.name, &person.age, &person.salary) == 3) {

copy\_element(&person, &data[i]);

i++;

}

fclose(pFile);

return true;

}

void Update\_1(struct PERSONAL \*personal1, struct PERSONAL\* personal2) {

for (int i = 0; i < 20; i++) {

copy\_element(&personal1[i], &personal2[i]);

if ((personal1[i].age / 10) \* 10 == 40) {

personal2[i].salary = personal1[i].salary \* 1.1;

}

}

}

void Update\_2(struct PERSONAL\* personal2, struct PERSONAL\* personal3) {

for (int i = 0; i < 20; i++) {

copy\_element(&personal2[i], &personal3[i]);

if ((personal2[i].age / 10) \* 10 == 30) {

personal3[i].salary = personal2[i].salary \* 1.2;

}

}

}

void Compare\_element(struct PERSONAL\* personal1, struct PERSONAL\* personal3) {

for (int i = 0; i < 20; i++) {

printf("%8s %d %.1lf -> %.1lf", personal1[i].name, personal1[i].age, personal1[i].salary, personal3[i].salary);

if (personal1[i].salary != personal3[i].salary)

printf(" Different\n");

else

printf("\n");

}

}

int main() {

char fname[50] = "salary\_v1.txt";

char fname\_2[30] = "salary\_v2.txt";

char fname\_3[30] = "salary\_v3.txt";

read\_file(fname,personal\_1);

Update\_1(personal\_1, temp);

write\_file(fname\_2, temp);

read\_file(fname\_2,personal\_2);

Update\_2(personal\_2, personal\_3);

write\_file(fname\_3, personal\_3);

Compare\_element(personal\_1,personal\_3);

return 0;

}

텍스트이(가) 표시된 사진

자동 생성된 설명