

## 박기홍 5일차 과제

### 1. HW\_001

[소스코드]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

// 학년 구조체 선언
typedef struct _Student {

    int schoolYear;
    double grade;
    char name[20];
}Student;

int main() {

    // 변수 선언하기.
    int personMax = 5;
    int i, j;

    // 구조체 정의하기
    Student student[5];
    Student temp; //이름 변경(SWAP)을 위한 구조체 변수 선언하기.
    //-> 입력된 학생의 수는 다섯 명이 한계이므로,
    // student변수의 index-5를 사용해도 되지만, SWAP이라는 용도를 나타내기 위해
    구조체에 대한 개별 변수를 선언함.

    // 형식 입출력하기.
    printf("입력 : ");
    // Name을 Null로 초기화 하기 (이름 변경을 위함)
    for (i = 0; i < personMax; i++) {
        for (int k = 0; k < 20; k++) {
            student[i].name[k] = '\0';
        }
    }
    for (i = 0; i < personMax; i++) {
        if (i != 0) {
            printf(" ");
        }
        scanf("%d %lf %s", &student[i].schoolYear, &student[i].grade,
&student[i].name);
    }

    // 정렬하기 (오름차순)
    for (i = 0; i < personMax; i++) {
        for (j = i; j < personMax; j++) {

            // 정렬 조건 1: 학년 기준으로 정렬하기 ,
```

정의하기.

```
if (student[i].schoolYear > student[j].schoolYear) {
    // 학년, 학점, 이름 변경을 위한 임시 변수 선언 및

    int tempSchoolYear;
    double tempGrade;

    // 학년 위치 변경하기
    tempSchoolYear = student[i].schoolYear;
    student[i].schoolYear = student[j].schoolYear;
    student[j].schoolYear = tempSchoolYear;

    // 학점 위치 변경하기
    tempGrade = student[i].grade;
    student[i].grade = student[j].grade;
    student[j].grade = tempGrade;

    // 이름 위치 변경하기
    for (int k = 0; student[i].name[k] != 0 ||
student[j].name[k] != 0; k++) {
        temp.name[k] = student[i].name[k];
        student[i].name[k] = student[j].name[k];
        student[j].name[k] = temp.name[k];
    }
}
```

정의하기.

```
}else if (student[i].schoolYear == student[j].schoolYear) {
    if (student[i].grade > student[j].grade) {
        // 학년, 학점, 이름 변경을 위한 임시 변수 선언 및

        int tempSchoolYear;
        double tempGrade;

        // 학년 위치 변경하기
        tempSchoolYear = student[i].schoolYear;
        student[i].schoolYear = student[j].schoolYear;
        student[j].schoolYear = tempSchoolYear;

        // 학점 위치 변경하기
        tempGrade = student[i].grade;
        student[i].grade = student[j].grade;
        student[j].grade = tempGrade;

        // 이름 위치 변경하기
        for (int k = 0; student[i].name[k] != 0 ||
student[j].name[k] != 0; k++) {
            temp.name[k] = student[i].name[k];
            student[i].name[k] = student[j].name[k];
            student[j].name[k] = temp.name[k];
        }
    }

    }else if (student[i].grade == student[j].grade) {
        // 이름의 각 번째마다 순서 비교하기
        for (int n = 0; n < personMax; n++) {
```

```

student[j].name[n]) {
    // 학년, 학점, 이름 변경을 위한
    int tempSchoolYear;
    double tempGrade;

    // 학년 위치 변경하기
    tempSchoolYear =
        student[i].schoolYear;
    student[i].schoolYear =
        student[j].schoolYear;
    tempSchoolYear =
        student[j].schoolYear;

    // 학점 위치 변경하기
    tempGrade = student[i].grade;
    student[i].grade =
        student[j].grade;
    student[j].grade = tempGrade;

    // 이름 위치 변경하기
    for (int k = 0;
        student[i].name[k] != 0 || student[j].name[k] != 0; k++) {
        temp.name[k] =
            student[i].name[k];
        student[i].name[k] =
            student[j].name[k];
        student[j].name[k] =
            temp.name[k];
    }
}

}

}

}

}

// 최종 결과 값 출력하기.
printf("\n출력 : ");
for (i = 0; i < personMax; i++) {
    if (i != 0) {
        printf(" ");
    }
    printf("%d %.1lf %s\n", student[i].schoolYear, student[i].grade,
&student[i].name);
}

return 0;
}

```

[실행결과]

### Test Case #1

```
Microsoft Visual Studio 디버그 콘솔
입력 : 4 4.5 kwon
       1 3.2 jo
       3 2.3 yoon
       3 1.6 koh
       3 2.3 park

출력 : 1 3.2 jo
       3 1.6 koh
       3 2.3 park
       3 2.3 yoon
       4 4.5 kwon

C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\x64\Debug\HW_20240705.exe
(프로세스 8512개)이(가) 종료되었습니다(코드: 0x0).
이 창을 닫으려면 아무 키나 누르세요...
```

### Test Case #2

```
Microsoft Visual Studio 디버그 콘솔
입력 : 3 1.5 ASIMO
       1 4.5 HUBO
       2 3.9 TOBOT
       4 4.5 NanoList
       4 4.4 ROBO

출력 : 1 4.5 HUBO
       2 3.9 TOBOT
       3 1.5 ASIMO
       4 4.4 ROBO
       4 4.5 NanoList

C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\x64\Debug\HW_20240705.exe
(프로세스 4904개)이(가) 종료되었습니다(코드: 0x0).
이 창을 닫으려면 아무 키나 누르세요...
```

### Test Case #3

```
Microsoft Visual Studio 디버그 콘솔
입력 : 1 1.2 Happy
      3 4.5 BlackWidow
      3 3.9 Captin
      4 4.5 IronMan
      4 4.4 SpiderMan

출력 : 1 1.2 Happy
      3 3.9 Captin
      3 4.5 BlackWidow
      4 4.4 SpiderMan
      4 4.5 IronMan

C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\x64\Debug\HW_20240705.exe
(프로세스 12448개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

## 2. HW\_002

[소스코드]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

typedef struct _position {
    int posX;
    int posY;
    int posTotal; // 가장 거리가 먼 좌표를 저장하기 위한 변수
}Position;

int main() {
    // 변수 생성하기.
    int inputNum = 0;
    int i;

    // 형식 입출력받기.
    printf("입력 : ");
    scanf("%d", &inputNum);

    int* pArrPosX = (int*)calloc(inputNum, sizeof(int));
    int* pArrPosY = (int*)calloc(inputNum, sizeof(int));

    // 포인터 개념을 사용하여 동적할당된 값을 구조체에 저장하기 위한 선언.
```

```

Position* position = calloc(inputNum, sizeof(Position));

for (i = 0; i < inputNum; i++) {
    printf(" ");
    scanf("%d %d", &pArrPosX[i], &pArrPosY[i]);
}

// 동적할당된 값을 구조체에 대입하기.
for (i = 0; i < inputNum; i++) {
    position[i].posX = pArrPosX[i];
    position[i].posY = pArrPosY[i];
    position[i].posTotal = (pArrPosX[i] * pArrPosX[i]) + (pArrPosY[i] *
pArrPosY[i]);
}

// 입력된 좌표 중 가장 멀리 있는 좌표를 찾아내기.
int highPosX = position[0].posX;
int highPosY = position[0].posY;
int highPosTot = position[0].posTotal;
for (i = 0; i < inputNum; i++) {
    for (int j = 0; j < inputNum; j++) {
        if (highPosTot < position[j].posTotal) {
            highPosX = position[j].posX;
            highPosY = position[j].posY;
            highPosTot = position[j].posTotal;
        }
    }
}

double distance = 0.0;
// 다른 좌표의 거리 총합 구하기.
for (i = 0; i < inputNum; i++) {
    if (position[i].posTotal != highPosTot) {
        position[i].posTotal = ((highPosX - position[i].posX) * (highPosX
- position[i].posX) + (highPosY - position[i].posY) * (highPosY - position[i].posY));
        distance = distance + sqrt((double)position[i].posTotal);
    }
}

// Debugging: Number는 임시로 출력.
printf("\n출력 : 가장 거리가 먼 좌표는 (%d, %d)이며, 다른 좌표의 거리 총합은
약 %.1lf입니다.", highPosX, highPosY, distance);

free(pArrPosX);
free(pArrPosY);
free(position);

return 0;
}

```

[실행결과]

### Test Case #1

```
Microsoft Visual Studio 디버그 콘솔
입력 : 5
      0 0
      1 1
      2 3
      2 1
      7 7

출력 : 가장 거리가 먼 좌표는 (7, 7)이며, 다른 좌표의 거리 총합은 약 32.6입니다.
C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\64\Debug\HW_20240705.exe
(프로세스 28668개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

### Test Case #2

```
Microsoft Visual Studio 디버그 콘솔
입력 : 3
      2 0
      1 3
      5 5

출력 : 가장 거리가 먼 좌표는 (5, 5)이며, 다른 좌표의 거리 총합은 약 10.3입니다.
C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\64\Debug\HW_20240705.exe
(프로세스 18608개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

### Test Case #3

```
Microsoft Visual Studio 디버깅 콘솔
입력 : 4
      2 3
      1 7
      5 5
      3 3

출력 : 가장 거리가 먼 좌표는 (1, 7)이며, 다른 좌표의 거리 총합은 약 8.6입니다.
C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\x64\Debug\HW_20240705.exe
(프로세스 17360개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

### 3. HW\_003

[소스코드]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //C언어 3일차 교육에서 배운 헤더파일.

typedef struct _something {

    char name[20];
    int cost;
}Something;

int main() {

    // 변수 선언하기.
    int inputNum, i, wantIdx = 1;
    // 형식 입출력하기.
    printf("입력 : ");
    scanf("%d", &inputNum);

    // 구조체에 동적할당하기.
    Something* something = calloc(inputNum, sizeof(char) * 20);

    // 동적할당한 값을 입력 받아 구조체에 대입하기.
    for (i = 0; i < inputNum; i++) {
```



```

        printf("      ");
        scanf("%s %d", &something[i].name, &something[i].cost);
        wantIdx++;
    }

    printf("      ");
    scanf("%s", &something[i].name);

    printf("\n출력 : ");

    int totalCost = 0;
    // 찾는 물건이 입력된 물건 리스트에 있는지 찾아보는 알고리즘.
    for (i = 0; i < inputNum; i++) {
        // 찾는 물건이 리스트에 있을 때,
        if (strcmp(something[i].name, something[inputNum].name) == 0) {
            totalCost = totalCost + something[i].cost;
        }
    }
    printf("%d", totalCost);
    // 동적할당 해제하기.
    free(something);

    return 0;
}

```

[실행결과]

Test Case #1

```

Microsoft Visual Studio 디버그 콘솔
입력 : 3
      Apple 2000
      Coffe 1000
      Apple 1500
      Orange

출력 : 0
C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\64\Debug\HW_20240705.exe
(프로세스 18448개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...

```

Test Case #2

```
Microsoft Visual Studio 디버그 콘솔
입력 : 5
    Apple 2000
    Coffe 1000
    Apple 1500
    Tea 300
    Apple 200
    Apple

출력 : 3700
C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\HW_20240705.exe
(프로세스 14532개)이(가) 종료되었습니다(코드: -1073740940개).
이 창을 닫으려면 아무 키나 누르세요...
```

### Test Case #3

```
Microsoft Visual Studio 디버그 콘솔
입력 : 4
    Apple 2000
    WaterMelon 1000
    Banana 1500
    Banana 3000
    Banana

출력 : 4500
C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\HW_20240705.exe
(프로세스 21944개)이(가) 종료되었습니다(코드: -1073741819개).
이 창을 닫으려면 아무 키나 누르세요...
```

### 4. HW\_004

#### [소스코드]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
```

```

typedef struct {
    int year, month, day;
}DATE;

typedef struct {
    int hour, min, sec;
}TIME;

typedef struct {
    DATE Date;
    TIME Time;
}TIMESTAMP;

// 두 개의 TimeStamp를 비교하는 함수.
int comparingTimestamp(TIMESTAMP stamp1, TIMESTAMP stamp2, int level);

int main() {
    // 구조체 선언하기.
    TIMESTAMP timeStamp1, timeStamp2;
    // 계산된 시간, 분, 초를 저장하는 변수.
    int resultHour = 0, resultMin = 0, resultSec = 0;

    printf("입력 : ");
    scanf("%d %d %d %d %d %d", &timeStamp1.Date.year, &timeStamp1.Date.month,
&timeStamp1.Date.day, &timeStamp1.Time.hour, &timeStamp1.Time.min, &timeStamp1.Time.sec);
    printf("      ");
    scanf("%d %d %d %d %d %d", &timeStamp2.Date.year, &timeStamp2.Date.month,
&timeStamp2.Date.day, &timeStamp2.Time.hour, &timeStamp2.Time.min, &timeStamp2.Time.sec);

    /* 계산 알고리즘.
    * 1. 날짜 비교하기.
    * 날짜(year, month, day)를 각각 비교하기.
    * 1) year 비교하기. if) year가 같다면, month 비교하기.
    * 2) month 비교하기. if) month가 같다면, day 비교하기.
    * 3) day 비교하기. if) day가 같다면, hour 비교하기.
    * 4) hour 비교하기. if) hour가 같다면, min 비교하기.
    * 5) min 비교하기. if) min이 같다면, sec 비교하기.
    * 6) sec 비교하기. if) sec가 같다면, '0시 0분 0초' 출력하기.
    * 큰 값은 더 뒤에 있는 날짜, 작은 값은 큰 값보다 앞에 있는 날짜.
    * 큰 값에서 작은 값을 빼야 하는 것.
    */

    /* 비교 레벨 선언하기.
    * 0 : year 비교
    * 1 : month 비교
    * 2 : day 비교
    * 3 : hour 비교
    * 4 : min 비교
    * 5 : sec 비교

```

```

*/
int compareLevel = 0;

if (timeStamp1.Date.year > timeStamp2.Date.year) {
    comparingTimestamp(timeStamp1, timeStamp2, compareLevel);
}else if (timeStamp1.Date.year == timeStamp2.Date.year){
    compareLevel++;
    if (timeStamp1.Date.month > timeStamp2.Date.month) {
        comparingTimestamp(timeStamp1, timeStamp2, compareLevel);
    }else if (timeStamp1.Date.month == timeStamp2.Date.month){
        compareLevel++;
        if (timeStamp1.Date.day > timeStamp2.Date.day) {
            comparingTimestamp(timeStamp1, timeStamp2, compareLevel);
        }else if (timeStamp1.Date.day == timeStamp2.Date.day) {
            compareLevel++;
            if (timeStamp1.Time.hour > timeStamp2.Time.hour) {
                comparingTimestamp(timeStamp1, timeStamp2,
compareLevel);
            }else if (timeStamp1.Time.hour == timeStamp2.Time.hour){
                compareLevel++;
                if (timeStamp1.Time.min > timeStamp2.Time.min) {
                    comparingTimestamp(timeStamp1, timeStamp2,
compareLevel);
                }else if (timeStamp1.Time.min ==
timeStamp2.Time.min){
                    compareLevel++;
                    if (timeStamp1.Time.sec >
timeStamp2.Time.sec) {
                        comparingTimestamp(timeStamp1,
timeStamp2, compareLevel);
                    }else{
                        comparingTimestamp(timeStamp2,
timeStamp1, compareLevel);
                    }
                }else {
                    comparingTimestamp(timeStamp2, timeStamp1,
compareLevel);
                }
            }else {
                comparingTimestamp(timeStamp2, timeStamp1,
compareLevel);
            }
        }else {
            comparingTimestamp(timeStamp2, timeStamp1, compareLevel);
        }
    }else {
        comparingTimestamp(timeStamp2, timeStamp1, compareLevel);
    }
}
}

```

```

        return 0;
    }

    int comparingTimestamp(TIMESTAMP stamp1, TIMESTAMP stamp2, int level) {
        // 계산을 위한 변수 선언하기.
        int tempYear = 0, tempMonth = 0, tempDay = 0, tempHour = 0, tempMin = 0, tempSec =
0;

        // 윤년임을 파악하기 위한 변수
        int isLeapYear = 0;

        // 나중에 남은 year, month, day를 hour로 변경하기 위해서 윤년임을 저장하는
알고리즘.
        if (((stamp1.Date.year % 4 == 0 && stamp1.Date.year % 100 != 0) ||
stamp1.Date.year % 400 == 0)) {
            isLeapYear = 1;
        }else{
            isLeapYear = 0;
        }

        // 차이 계산하기.
        tempYear = stamp1.Date.year - stamp2.Date.year;

        // 늦은 시각과 빠른 시각의 차가 0 미만일 때, year에서 1을 빼고 month를 추가한다.
        if (stamp1.Date.month - stamp2.Date.month < 0) {
            tempYear--;
            stamp1.Date.month = stamp1.Date.month + 12; // 1년을 제외하고, 12개월을
추가함.
        }
        tempMonth = stamp1.Date.month - stamp2.Date.month;

        // 늦은 시각과 빠른 시각의 차가 0 미만일 때, month에서 1을 빼고 day를 추가한다.
        if (stamp1.Date.day - stamp2.Date.day < 0) {
            // 짝수 달인지, 홀수 달인지 체크하기.
            if(stamp1.Date.day % 2 == 0){ // 짝수 달이라면,
                // 윤년 검증하기. -> 2월달이고, 윤년이라면 29일을 추가.
                if (stamp1.Date.month == 2 && ((stamp1.Date.year % 4 == 0 &&
stamp1.Date.year % 100 != 0) || stamp1.Date.year % 400 == 0)) {
                    tempMonth--;
                    stamp1.Date.day = stamp1.Date.day + 29;
                }else if(stamp1.Date.day == 2){ // 윤년이 아닌 2월달이라면 28일을
추가.
                    tempMonth--;
                    stamp1.Date.day = stamp1.Date.day + 28;
                }else{ // 2월달 제외 짝수달의 day 처리.
                    tempMonth--;
                    stamp1.Date.day = stamp1.Date.day + 30;
                }
            }else{ // 홀수 달이라면,
                tempMonth--;
                stamp1.Date.day = stamp1.Date.day + 31;
            }
        }
    }
}

```

```

tempDay = stamp1.Date.day - stamp2.Date.day;

// 늦은 시각과 빠른 시각의 차가 0 미만일 때, day에서 1을 빼고 hour를 추가한다.
if (stamp1.Time.hour - stamp2.Time.hour < 0) {
    tempDay--;
    stamp1.Time.hour = stamp1.Time.hour + 24;
}
tempHour = stamp1.Time.hour - stamp2.Time.hour;

// 늦은 시각과 빠른 시각의 차가 0 미만일 때, hour에서 1을 빼고 min을 추가한다.
if (stamp1.Time.min - stamp2.Time.min < 0) {
    tempHour--;
    stamp1.Time.min = stamp1.Time.min + 60;
}
tempMin = stamp1.Time.min - stamp2.Time.min;

// 늦은 시각과 빠른 시각의 차가 0 미만일 때, min에서 1을 빼고 sec을 추가한다.
if (stamp1.Time.sec - stamp2.Time.sec < 0) {
    tempMin--;
    stamp1.Time.sec = stamp1.Time.sec + 60;
}
tempSec = stamp1.Time.sec - stamp2.Time.sec;

// year, month, day를 hour로 환산하는 알고리즘.
// 1. year를 hour로 변환하기.
tempHour = tempHour + ((tempYear * 12) * 31) * 24;

// 2. month를 hour로 변환하기.
while (1) {
    if (tempMonth == 0) {
        break;
    }
    switch (tempMonth) {
    case 1:
        tempHour = tempHour + 31 * 24; tempMonth--; break;
    case 2:
        // 윤년일 때는 29일로 변환하여 계산하기.
        if (isLeapYear == 1) {
            tempHour = tempHour + 29 * 24; tempMonth--; break;
        } else { // 윤년이 아닐때는 28일로 변환하여 계산하기.
            tempHour = tempHour + 28 * 24; tempMonth--; break;
        }
    case 3:
        tempHour = tempHour + 31 * 24; tempMonth--; break;
    case 4:
        tempHour = tempHour + 30 * 24; tempMonth--; break;
    case 5:
        tempHour = tempHour + 31 * 24; tempMonth--; break;
    case 6:
        tempHour = tempHour + 30 * 24; tempMonth--; break;
    case 7:
        tempHour = tempHour + 31 * 24; tempMonth--; break;
    }
}

```

```

        case 8:
            tempHour = tempHour + 30 * 24; tempMonth--; break;
        case 9:
            tempHour = tempHour + 31 * 24; tempMonth--; break;
        case 10:
            tempHour = tempHour + 30 * 24; tempMonth--; break;
        case 11:
            tempHour = tempHour + 31 * 24; tempMonth--; break;
        case 12:
            tempHour = tempHour + 30 * 24; tempMonth--; break;
        default:
            break;
    }
}

// 3. day를 hour로 변환하기.
tempHour = tempHour + tempDay * 24;

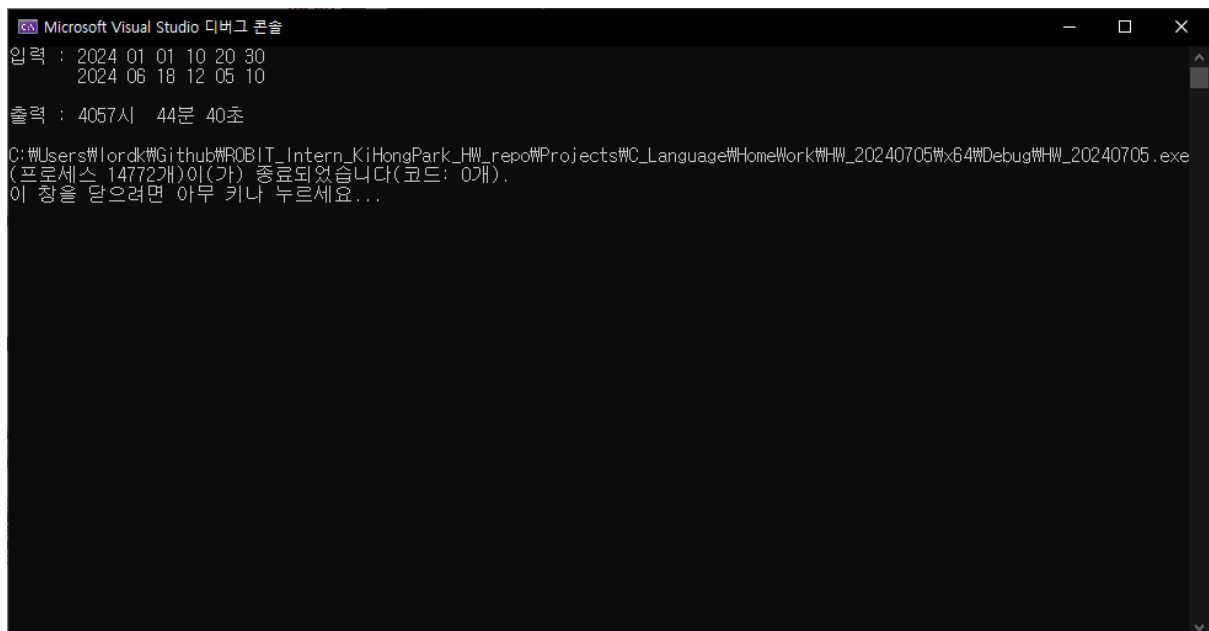
printf("\n출력 : %d시 %d분 %d초\n", tempHour, tempMin, tempSec);

return 0;
}

```

[실행결과]

Test Case #1



The screenshot shows a Windows command prompt window titled "Microsoft Visual Studio 디버그 콘솔". It displays the following text:

```

입력 : 2024 01 01 10 20 30
      2024 06 18 12 05 10

출력 : 4057시 44분 40초

C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\64\Debug\HW_20240705.exe
(프로세스 14772개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...

```

Test Case #2

```
Microsoft Visual Studio 디버깅 콘솔
입력 : 2024 05 01 10 20 30
       2024 06 18 12 05 10

출력 : 1153시 44분 40초

C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\64\Debug\HW_20240705.exe
(프로세스 11032개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

### Test Case #3

```
Microsoft Visual Studio 디버깅 콘솔
입력 : 2024 03 19 10 20 30
       2022 03 19 10 20 30

출력 : 17856시 0분 0초

C:\Users\lordk\Github\ROBIT_Intern_KiHongPark_HW_repo\Projects\C_Language\HomeWork\HW_20240705\64\Debug\HW_20240705.exe
(프로세스 8572개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```