

EC C언어 스터디

-6강-



구조체



파일 입출력





구조체

```
#include <stdio.h>
#include <math.h>

int main()
{
    int a_x = 1, a_y = 2;
    int b_x = 4, b_y = 6;

    printf("%f", sqrt((b_x - a_x) * (b_x - a_x) + (b_y - a_y) * (b_y - a_y)));
}
```

```
#include <stdio.h>
#include <math.h>

int main()
{
    int point_x[2] = {1, 4};
    int point_y[2] = {2, 6};
    printf("%f", sqrt((point_x[1] - point_x[0]) * (point_x[1] - point_x[0]) +
(point_y[1] - point_y[0]) * (point_y[1] - point_y[0])));
}
```

```
#include <stdio.h>

int main()
{
    int a_id = 18101290;
    int a_age = 21;
    char a_major[30] = "Computer Engineering";
    char a_name[20] = "InSeoHwang";

    int b_id = 18101289;
    int b_age = 21;
    char b_major[30] = "Computer Engineering";
    char b_name[20] = "HaJiMwo";

    printf("a학생의 학번 : %d\n", a_id);
    printf("a학생의 나이 : %d\n", a_age);
    printf("a학생의 전공 : %s\n", a_major);
    printf("a학생의 이름 : %s\n", a_name);
    printf("\n");
    printf("b학생의 학번 : %d\n", b_id);
    printf("b학생의 나이 : %d\n", b_age);
    printf("b학생의 전공 : %s\n", b_major);
    printf("b학생의 이름 : %s\n", b_name);
}
```

```
struct 구조체이름  
{  
    구조체 변수 선언  
};
```

```
struct Point  
{  
    int x, y;  
};
```

```
#include <stdio.h>
#include <math.h>

struct Point
{
    int x, y;
};

int main()
{
    struct Point a;
    // struct 구조체이름 구조체변수의이름 으로 선언합니다.
    a.x = 1;
    a.y = 2;
    // 구조체 멤버는 . 을 통해 접근할 수 있습니다.
    printf("a.x = %d , a.y = %d", a.x, a.y);
}
```



```
#include <stdio.h>
#include <math.h>

struct Point
{
    int x, y;
};

int main()
{
    struct Point a = { 1, 2 }, b = { 4, 6 };
    // 이런식으로 초기화 해줄 수 있습니다.
    printf("%f", sqrt((b.x - a.x) * (b.x - a.x) + (b.y - a.y) * (b.y - a.y)));
}
```

```
struct Student
{
    int id, age;
    char major[32], name[20];
};

int main()
{
    struct Student a = { 18101290, 21, "Computer Engineering", "InSeoHwang" },
        b = { 18101289, 21, "Computer Engineering", "a" };
    //문자열도 초기화가능

    strcpy(b.name, "HaJiMwo");
    // 문자열에 값을 넣기 위해선 strcpy(대상, 넣을 문자열) (string_copy약자) 함수 사용
    printf("a학생의 학번 : %d\n", a.id);
    printf("a학생의 나이 : %d\n", a.age);
    printf("a학생의 전공 : %s\n", a.major);
    printf("a학생의 이름 : %s\n", a.name);
    printf("\n");
    printf("b학생의 학번 : %d\n", b.id);
    printf("b학생의 나이 : %d\n", b.age);
    printf("b학생의 전공 : %s\n", b.major);
    printf("b학생의 이름 : %s\n", b.name);
}
```

```
struct 구조체이름 {  
    자료형 멤버이름;  
} 변수;
```

```
struct Student  
{  
    int id, age;  
    char major[32], name[20];  
}a;
```

```
#include <stdio.h>
#include <math.h>

struct Point
{
    int x, y;
}p[2];

int main()
{
    p[0].x = 1, p[0].y = 2;
    p[1].x = 4, p[1].y = 6;
    printf("%f", sqrt((p[1].x - p[0].x) * (p[1].x - p[0].x) + (p[1].y - p[0].y) * (p[1].y - p[0].y)));
}
```

```
typedef long long int ll;  
  
int main()  
{  
    ll a;  
    long long int b;  
    // a와 b는 같은 타입  
}
```

```
typedef struct pnt
{
    int x, y;
}Point;
// struct pnt의 별명을 point로 해줌

int main()
{
    Point a = { 1,2 }, b = { 4, 6 };
    // Point만 써도 구조체 변수 선언 가능 Point를 자료형처럼 사용함
    printf("%f", sqrt((b.x - a.x) * (b.x - a.x) + (b.y - a.y) * (b.y - a.y)));
}
```

```
typedef struct
{
    int x, y;
}Point;

int main()
{
    Point a = { 1,2 }, b = { 4, 6 };
    printf("%f", sqrt((b.x - a.x) * (b.x - a.x) + (b.y - a.y) * (b.y - a.y)));
}
```

```
int main()
{
    int a=1, b=2;
    b = a;
}
```


이러한 구조체 복사 함수는 `memcpy` 함수를 사용하여 주소, 대상의 크기);

```
#include <stdio.h>
#include <string.h>

struct Student
{
    int id, age;
    char major[32], name[20];
};

int main()
{
    struct Student a = { 18101290, 21, "Computer Engineering", "InSeoHwang" }, b;

    memcpy(&b, &a, sizeof(a));

    printf("a학생의 학번 : %d\n", a.id);
    printf("a학생의 나이 : %d\n", a.age);
    printf("a학생의 전공 : %s\n", a.major);
    printf("a학생의 이름 : %s\n", a.name);
    printf("\n");
    printf("b학생의 학번 : %d\n", b.id);
    printf("b학생의 나이 : %d\n", b.age);
    printf("b학생의 전공 : %s\n", b.major);
    printf("b학생의 이름 : %s\n", b.name);
    printf("\n");
}
```

```
struct Student
{
    int id, age;
    char major[32], name[20];
};

int main()
{
    struct Student a = { 18101290, 21, "Computer Engineering", "InSeoHwang" }, b;

    memcpy(&b, &a, sizeof(a));

    printf("a학생의 학번 : %d\n", a.id);
    printf("a학생의 나이 : %d\n", a.age);
    printf("a학생의 전공 : %s\n", a.major);
    printf("a학생의 이름 : %s\n", a.name);
    printf("\n");
    printf("b학생의 학번 : %d\n", b.id);
    printf("b학생의 나이 : %d\n", b.age);
    printf("b학생의 전공 : %s\n", b.major);
    printf("b학생의 이름 : %s\n", b.name);
}
```

```
a학생의 학번 : 18101290
a학생의 나이 : 21
a학생의 전공 : Computer Engineering
a학생의 이름 : InSeoHwang
:
b학생의 학번 : 18101290
b학생의 나이 : 21
b학생의 전공 : Computer Engineering
b학생의 이름 : InSeoHwang
```

```
typedef struct
{
    int* p;
}Data;

int main()
{
    int n = 1;
    Data a = { &n };
    Data* sp = &a;
    printf("%p\n", sp->p); // = a.p = &n
    //구조체 포인터를 통해 접근할 때는 화살표를 사용합니다.
    //또는 (*sp).p
    printf("%d\n", *sp->p); // = *a.p = n
}
```

```
struct 구조체이름 *포인터이름 = malloc(sizeof(struct 구조체이름));
```

```
typedef struct Student
{
    int id, age;
    char major[32], name[20];
}Student;

int main()
{
    Student* p = (Student*)malloc(sizeof(Student));

    printf("학생의 학번을 입력하세요");
    scanf("%d", &student->id);
    printf("학생의 나이를 입력하세요");
    scanf("%d", &student->age);
    printf("학생의 전공을 입력하세요");
    scanf("%s", student->major);
    printf("학생의 이름을 입력하세요");
    scanf("%s", student->name);

    printf("학생의 학번 : %d\n", p->id);
    printf("학생의 나이 : %d\n", p->age);
    printf("학생의 전공 : %s\n", p->major);
    printf("학생의 이름 : %s\n", p->name);
}
```

```
학생의 학번을 입력하세요 : 18101290
학생의 나이를 입력하세요 : 21
학생의 전공을 입력하세요 : CE
학생의 이름을 입력하세요 : his
학생의 학번 : 18101290
학생의 나이 : 21
학생의 전공 : CE
학생의 이름 : his
```

```

#include <stdio.h>
#define MAX_SIZE 100

typedef struct Student
{
    int id, age;
    char major[32], name[20];
}Student;

Student* arr[MAX_SIZE];
int student_num;

void input_data(Student* student)
{
    printf("학번을 입력하세요 : ");
    scanf("%d", &student->id);
    printf("나이를 입력하세요 : ");
    scanf("%d", &student->age);
    printf("전공을 입력하세요 : ");
    scanf("%s", student->major);
    printf("이름을 입력하세요 : ");
    scanf("%s", student->name);
}

void printf_data(Student student)
{
    printf("학번 : %d\n", student.id);
    printf("나이 : %d\n", student.age);
    printf("전공 : %s\n", student.major);
    printf("이름 : %s\n", student.name);
}

```

```

void allocate_student()
{
    printf("\n");
    for (int i = 0; i < student_num; i++)
    {
        arr[i] = (Student*)malloc(sizeof(Student));
        printf("%d번째 학생의 정보 입력\n", i + 1);
        input_data(arr[i]);
        printf("%d번째 입력 완료!\n\n", i + 1);
    }
}

int main()
{
    printf("입력할 학생의 수(최대 %d명) : ", MAX_SIZE);
    scanf("%d", &student_num);
    allocate_student();

    for (int i = 0; i < student_num; i++)
    {
        printf("--%d번째 학생의 정보-----\n", i + 1);
        printf_data(*arr[i]);
        printf("-----\n\n");
    }
}

```

입력할 학생의 수(최대 100명) : 2

1번째 학생의 정보 입력
 학번을 입력하세요 : 18101290
 나이를 입력하세요 : 21
 전공을 입력하세요 : 1
 이름을 입력하세요 : a
 1번째 입력 완료!

2번째 학생의 정보 입력
 학번을 입력하세요 : 123456
 나이를 입력하세요 : 321
 전공을 입력하세요 : hi
 이름을 입력하세요 : hello
 2번째 입력 완료!

--1번째 학생의 정보-----
 학번 : 18101290
 나이 : 21
 전공 : 1
 이름 : a

--2번째 학생의 정보-----
 학번 : 123456
 나이 : 321
 전공 : hi
 이름 : hello

```
struct A
{
    int a;
    struct A *p;
}
```

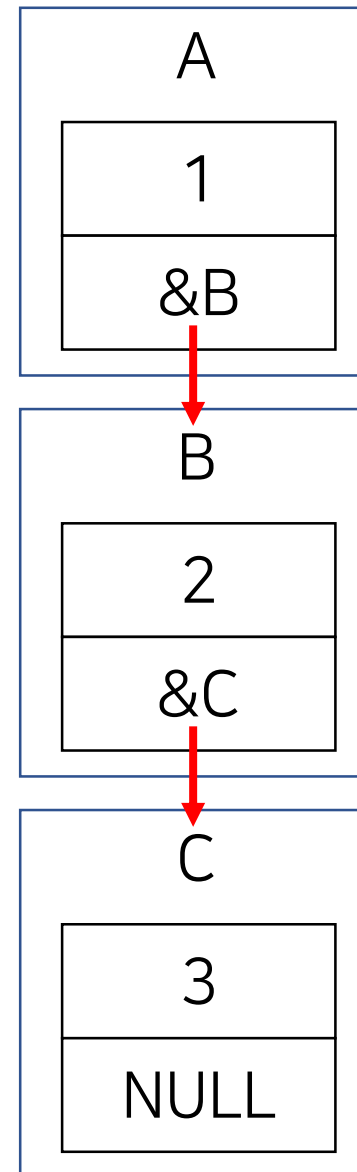
```
#include <stdio.h>

typedef struct Data
{
    int val;
    struct Data* next;
}Data;

int main()
{
    Data a, b, c;
    a.val = 1;
    a.next = &b;
    b.val = 2;
    b.next = &c;
    c.val = 3;
    c.next = NULL;

    for (Data *x = &a; x != NULL; x = x->next)
    {
        printf("%d ", x->val);
    }
}
```

1 2 3




```
typedef struct Data
{
    int val;
    struct Data* next;
}Data;

int main()
{
    Data *p = NULL, *start = NULL;

    int n;
    printf("연결리스트의 길이 : ");
    scanf("%d", &n);


    for (int i = 0; i < n; i++)
    {
        Data* temp = (Data*)malloc(sizeof(Data));
        printf("숫자 입력 : ");
        scanf("%d", &temp->val);
        temp->next = NULL;
        if (p != NULL) p->next = temp;
        else start = temp;
        p = temp;
    }

    for (Data* x = start; x != NULL; x = x->next)
    {
        printf("%d ", x->val);
    }
}
```

연결리스트의 길이	:	5
숫자 입력	:	1
숫자 입력	:	2
숫자 입력	:	3
숫자 입력	:	4
숫자 입력	:	5
1 2 3 4 5		

```
struct Data  
{  
    char c;  
    int n;  
}
```

```
#pragma pack(push, 1)    // 1바이트 크기로 정렬
struct Data
{
    char c;
    int n;
};
#pragma pack(pop)
```



파일 입출력

- 파일 열기 : `FILE* fp = fopen("test.txt", "w");`
- 파일에 쓰기 : `fprintf(fp, "helloworld %d", 1);`
- 파일에서 읽기 : `fscanf(fp, "%d", &n);`
- 파일 입출력 종료 : `fclose(fp);` // 꼭 해줘야 합니다.

```
int prime[100] = { 1,1 };

int main()
{
    for (int i = 2; i * i < 100; i++)
    {
        if (prime[i]) continue;
        for (int j = 2; i * j < 100; j++)
        {
            prime[i * j] = 1;
        }
    }

    FILE* fp = fopen("test.txt", "w");
    for (int i = 0; i < 100; i++)
    {
        if (!prime[i])
        {
            fprintf(fp, "%d ", i);
        }
    }
    fclose(fp);
}
```

test.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97



수고하셨습니다!