KG 아이티뱅크

C언어

VISUAL STUDIO

구조체

구조체

정의

- 여러 변수들을 묶어 그룹화하여 사용하는 사용자정의 자료형

구조체의 기본구조

```
struct    구조체이름
{
          자료형    멤버변수명;
          자료형    멤버변수명;
          자료형    멤버변수명;
          ……
};
```

```
struct    people
{
          char name[10];
          int age;
          char add[30];
};
```

구조체

<파일이름 : 01.구조체.c>

```
#include<stdio.h>

struct student {
	char name[10];
	int age;
	char address[20];
};

void main() {
	struct student s1 = {  " 홍길동 " , 29, "민락동" };
	printf("%s %d %s₩n", s1.name, s1.age, s1.address);
}
```

구조체명 뒤에 .을 찍으면 그 안의 변수들을 사용 가능

구조체 자료형 정의

<파일이름 : 02.구조체.c>

```
#include<stdio.h>

typedef struct student {
	char name[10];
	int age;
	char address[20];
}Student;

void main() {
	Student s1 = {  " 홍길동 " , 27, "민락동" };
	printf("%s %d %s₩n", s1.name, s1.age, s1.address);
}
```

typedef를 사용하여 다른 별칭을사용해도 구조체를 정의 가능

여러 구조체 변수

<파일이름 : 03.구조체.c>

```
#include<stdio.h>

typedef struct student {
	char name[10];
	int age;
	char address[20];
}Student;

void main() {
	Student s1 = {  " 홍길동 " , 29,  " 민락동 "  };
	Student s2 = {  " 김길동 " , 29, " 민락동 " };

	printf("%s %d %s₩n", s1.name, s1.age, s1.address);
	printf("%s %d %s₩n", s2.name, s2.age, s2.address);
}
```

구조체를 자료형의 묶음 이라고 생각하고 자료형과 똑같이 사용

구조체 업데이트

<파일이름 : 04.구조체.c>

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

typedef struct student {
	char name[10];
	int age;
	char address[20];
}Student;

void main() {
	Student s1 = {  " 홍길동", 29, "민락동" };
	printf("%s %d %s₩n", s1.name, s1.age, s1.address);

	strcpy(s1.name,  " 김길동");
	s1.age = 21;
	printf("%s %d %s₩n", s1.name, s1.age, s1.address);
}
```

구조체 변수명을 적고 . 안에 있는 변수명을 적은 후 수식 적용

구조체의 크기

- 구조체의 멤버중에서 가장 큰 자료형을 기본 단위로 설정하고 메모리 할당
- 자료형 별 메모리 할당 위치

자료형	크기	할당 위치
char	1byte	자유로이 메모리 주소 할당
short int	2byte	2의 배수로 메모리 주소 할당
int, float	4byte	4의 배수로 메모리 주소 할당
double	8byte	8의 배수로 메모리 주소 할당

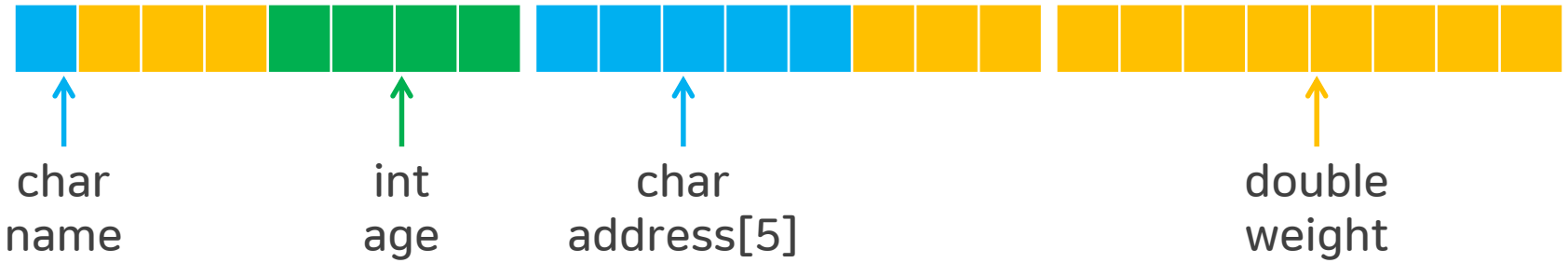
구조체의 크기

<파일이름 : 05.구조체.c>

```
#include<stdio.h>

typedef struct student {
	char name;
	int age;
	char address[5];
	double weight;
}Student;

void main() {
	Student s1;
	printf("%d₩n", sizeof(s1));
}
```

구조체 멤버 변수 중 가장 큰 double(8byte)를 기준으로 공간 할당

구조체의 크기



8byte를 기준으로 공간을 채워나감

구조체의 배열

<파일이름 : 06.구조체.c>

```
#include <stdio.h>

typedef struct member {
	char name[30];
	char 성별[6];
	int age;
	char phone_n[14];
} Member;

void main()
{
	Member arr_m[5] = { "홍길동", "man", 37, "010-1000-1000",
		"북길동", "man", 23, "010-2000-2000",
		"남길동", "woman", 26, "010-3000-3000",
		"서길동", "man", 45, "010-4000-4000",
		"동길동", "woman", 40, "010-5000-5000"
	};
	for (int i = 0; i < 5; i++) {
		printf("이름 : %3s, 성별 : %-5s, 나이 : %d, 전화번호 : %s₩n", arr_m[i].name, arr_m[i].성별,
		arr_m[i].age, arr_m[i].phone_n);
	}
}
```

구조체 배열을 만들고 멤버변수에 맞는 데이터 삽입

구조체의 배열

<파일이름 : 06.구조체.c>

	name	성별	age	phone_n
arr_m[0]	홍길동	man	37	010-1000-1000
arr_m[1]	북길동	man	16	010-2000-2000
arr_m[2]	남길동	woman	15	010-3000-3000
arr_m[3]	서길동	man	27	010-4000-4000
arr_m[4]	동길동	woman	42	010-5000-5000

구조체와 포인터

<파일이름 : 07.구조체.c>

```
#include <stdio.h>

typedef struct member {
	char name[30];
	char 성별[6];
	int age;
} Member;

void main()
{
	Member m1 = { "송진우", "man", 29 };
	Member* p_m1;
	p_m1 = &m1;
	//(*p_m1).age = 30;
	p_m1->age = 30;
	printf("%s %s %d ₩n", p_m1->name, p_m1->성별, p_m1->age);
}
```

(*p_m1).age와 p_m1->age는 같다

구조체와 함수

<파일이름 : 08.구조체.c>

```
#include <stdio.h>

typedef struct member {
        char name[30];
        int age;
} Member;

void Age_check(Member mm1) {
        if(mm1.age< 20)
                printf("미성년자 입니다. ₩n");
        else
                printf("성인 입니다. ₩n");
}

void main() {
        Member m1 = {  " 송진우", 29 };
        Age_check(m1);
}
```

함수의 매개변수 역시 Member형태로 선언

구조체와 함수

<파일이름 : 09.구조체.c>

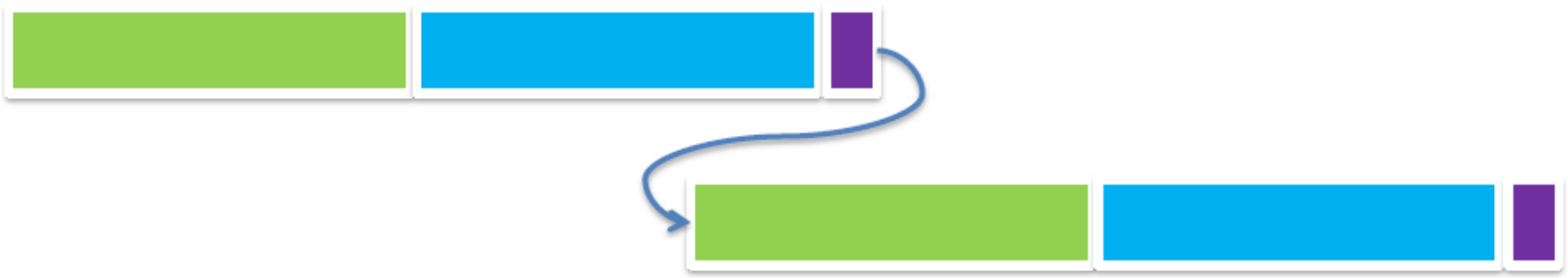
```
#include <stdio.h>

typedef struct member {
        char name[30];
        int age;
} Member;

void Age_check(Member* p_m1) {
        if(p_m1-> age< 20)
                printf("미성년자 입니다. ₩n");
        else
                printf("성인 입니다. ₩n");
}

void main() {
        Member m1 = {  " 송진우", 29 };
        Age_check(&m1);
}
```

주소를 매개변수로 받을 경우 Member* 형태로 선언

자기 참조 구조체

- 자기 참조 구조체(self-referential structure)
  - 자기 자신을 가리키는 포인터를 멤버로 가지는 구조체
  - 리스트(list)나 트리(tree)와 같은 자료 구조 구현에 사용

```
struct    people    // 자기 참조 구조체
{
          char name[10];
          int age;
          char add[30];
          struct  people* nextpeople
};
```



자기 참조 구조체

<파일이름 : 10.자기참조구조체.c>

```
#include <stdio.h>

typedef struct people {
        char name[10];
        int age;
        struct people* link;
} People;

void main() {
        People p_1 = {"Park", 22, NULL};
        People p_2 = {"Kim", 25, NULL};
        People p_3 = {"Lee", 18, NULL};

        p_1.link = &p_2;
        p_2.link = &p_3;
        p_3.link = &p_1;

        p_2.link->age = 30;
}
```