Chapter 9. Structures

May, 2016

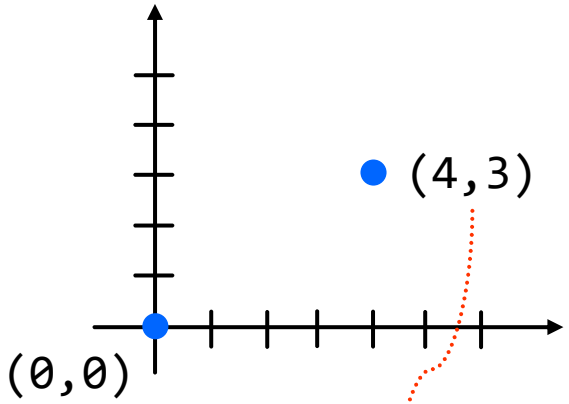
Seungjae Baek

Dept. of software
Dankook University
http://embedded.dankook.ac.kr/~baeksj

구조체의 개념 (1/4)

- 구조체: 다양한 종류의 데이터로 구성된 사용자 정의 데이터 타입
  - ✓ 복잡한 자료를 다루는 것을 편하게 해줌

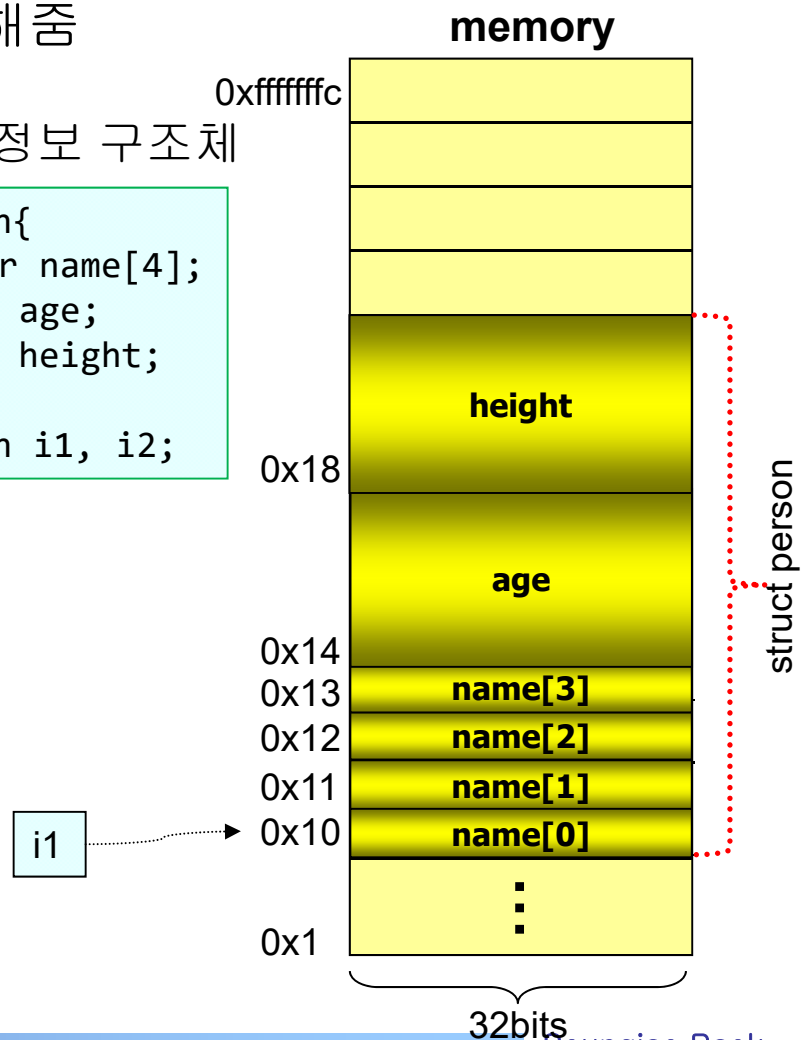
- 예 #1: 정수로 이루어진 x, y의 좌표




```
struct point {
      int x;
      int y;
}
```

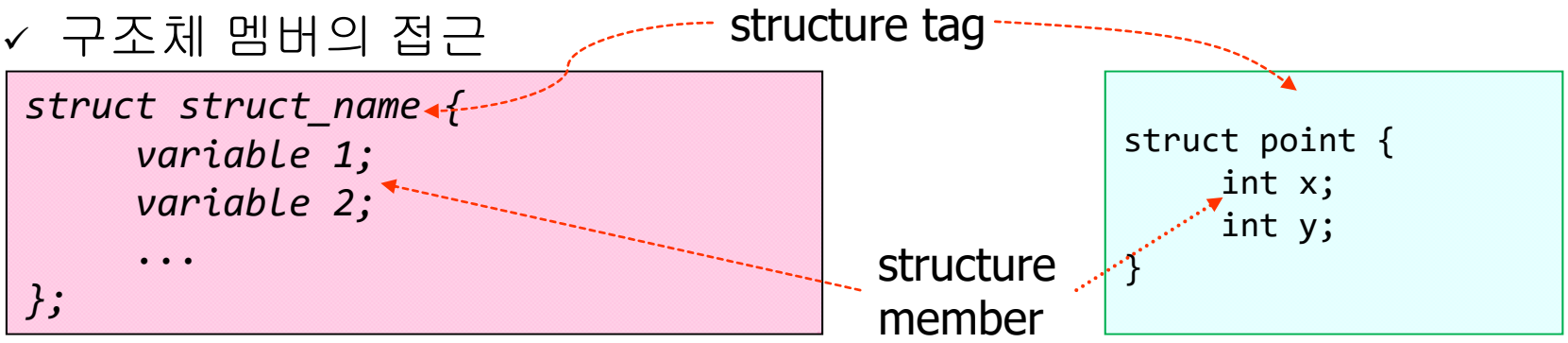
- 예 #2: 개인 정보 구조체

```
struct person{
          char name[4];
          int age;
          int height;
};
struct person i1, i2;
```



구조체의 개념 (2/4)

- 구조체의 형식
  - ✓ 구조체 멤버의 접근



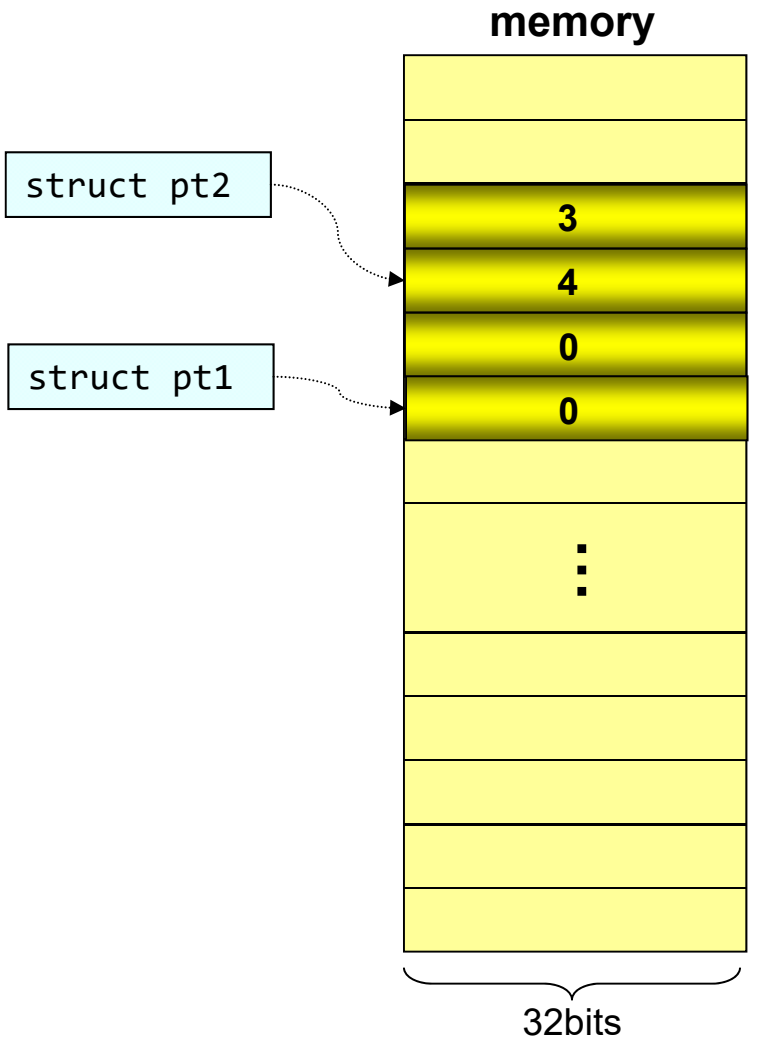
- 예제
  - ✓ 두 점의 기울기 구하기

```
#include <stdio.h>

struct point {
        int x;
        int y;
};

int main(void)
{
        double gradient;
        struct point pt1 = {0, 0};
        struct point pt2;
        pt2.x = 4;
        pt2.y = 3;

        printf("Point 1 (%d, %d)\n", pt1.x, pt1.y);
        printf("Point 2 (%d, %d)\n", pt2.x, pt2.y);
        gradient = (double)(pt2.y-pt1.y)/(pt2.x-pt1.x);
        printf("Gradient = %lf\n", gradient);

        return 0;
}
```

구조체의 개념 (3/4)

- 구조체의 메모리 구조

```
#include <stdio.h>

struct point {
    int x;
    int y;
};

int main()
{
    double gradient;
    struct point pt1 = {0, 0};
    struct point pt2;

    pt2.x = 4;
    pt2.y = 3;
    ...
```



구조체의 개념 (4/4)

- 구조체의 사용 예
  - ✓ 학생 구조체 선언 및 구조체 변수 초기화

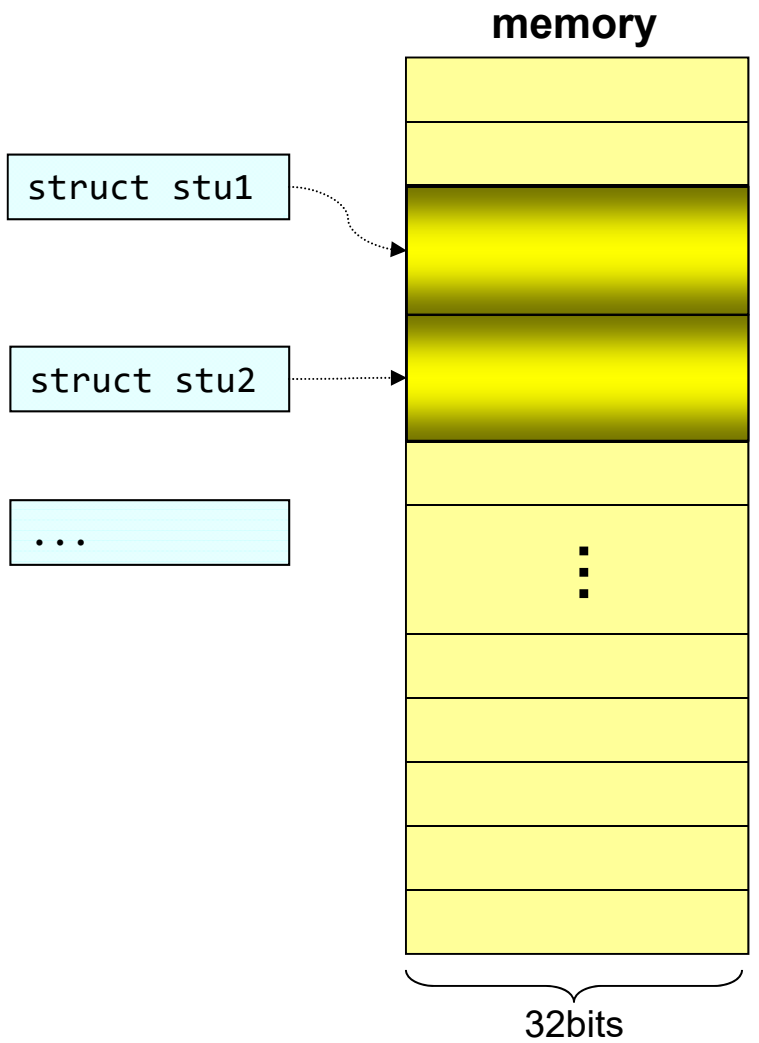
```
#include <stdio.h>
#include <string.h>

struct student {
    int id;
    char name[24];
    char addr[128];
    int zip;
};

int main()
{
    struct student stu1;
    struct student stu2
       = {20151234,
          "Taylor Swift",
          "Hollywood Blvd, USA",
          92013};
    printf("Student #2\n");
    printf(" ID   : %d\n", stu2.id);
    printf(" Name : %s\n", stu2.name);
    printf(" Addr : %s\n", stu2.addr);
    printf(" ZIP  : %d\n", stu2.zip);

    stu1.id = 20135678;
    strcpy(stu1.name, "Sam Smith");
    strcpy(stu1.addr, "Baker st. 207, UK");
    stu1.zip = 20002;

    printf("Student #2\n");
    printf(" ID   : %d\n", stu1.id);
    printf(" Name : %s\n", stu1.name);
    printf(" Addr : %s\n", stu1.addr);
    printf(" ZIP  : %d\n", stu1.zip);

    return 0;
}
```

구조체의 배열

- 구조체 연습문제
  - ✓ 학생 구조체를 이용하여 5명의 학생 데이터를 저장하려면?

Hint: 배열 사용

```
#include <stdio.h>
#include <string.h>
struct student {
        int id;
        char name[24];
        char addr[128];
        int zip;
};
struct student arr[5];

int main()
{
        arr[0].id = 123456;
        strcpy(arr[0].name, "Taylor Swift");
        strcpy(arr[0].addr, "Hollywood Blvd, USA");
        arr[0].zip = 92013;

        printf("Student #0\n");
        printf(" ID   : %d\n", arr[0].id);
        printf(" Name : %s\n", arr[0].name);
        printf(" Addr : %s\n", arr[0].addr);
        printf(" ZIP  : %d\n", arr[0].zip);
        …
        return 0;
}
```



구조체의 배열

구조체 배열 선언 및 초기화

```
        …
struct student {
        int id;
        char name[24];
        char addr[128];
        int zip;
};
struct student arr[5];

int main()
{
        arr[0].id = 123456;
        strcpy(arr[0].name, "Taylor Swift");
        strcpy(arr[0].addr, "Hollywood Blvd, USA");
        arr[0].zip = 92013;
        …
```

```
        …
struct student {
        int id;
        char name[24];
        char addr[128];
        int zip;
}arr[ ] = {
        123456, “Taylor Swift”, “Hollywood Blvd, USA“, 92013,
        …
        };
```

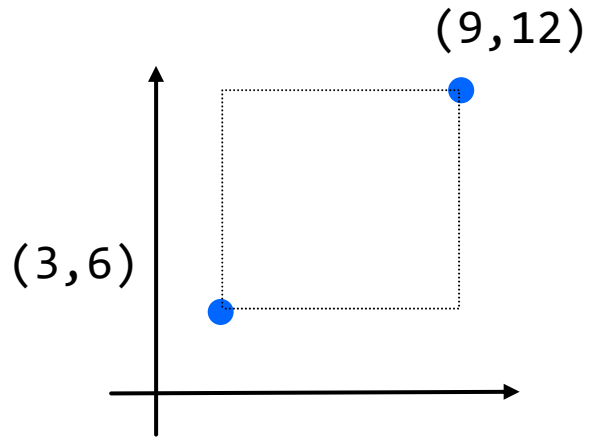
```
        …
struct student {
        int id;
        char name[24];
        char addr[128];
        int zip;
}arr[5];

int main()
{
        arr[0].id = 123456;
        strcpy(arr[0].name, "Taylor Swift");
        strcpy(arr[0].addr, "Hollywood Blvd, USA");
        arr[0].zip = 92013;
        …
```

```
        …
struct yourstudent {
        int id;
        char name[24];
        char addr[128];
        int zip;
}yourarr[] = {
        { 123456, "Taylor Swift", "Hollywood Blvd, USA", 92013},
        { 654321, "Sam Smith", "Baker st. 207, UK", 20002 },
        { 987654, "John Doe", "Unknown", }
};
         …
```

구조체의 중첩

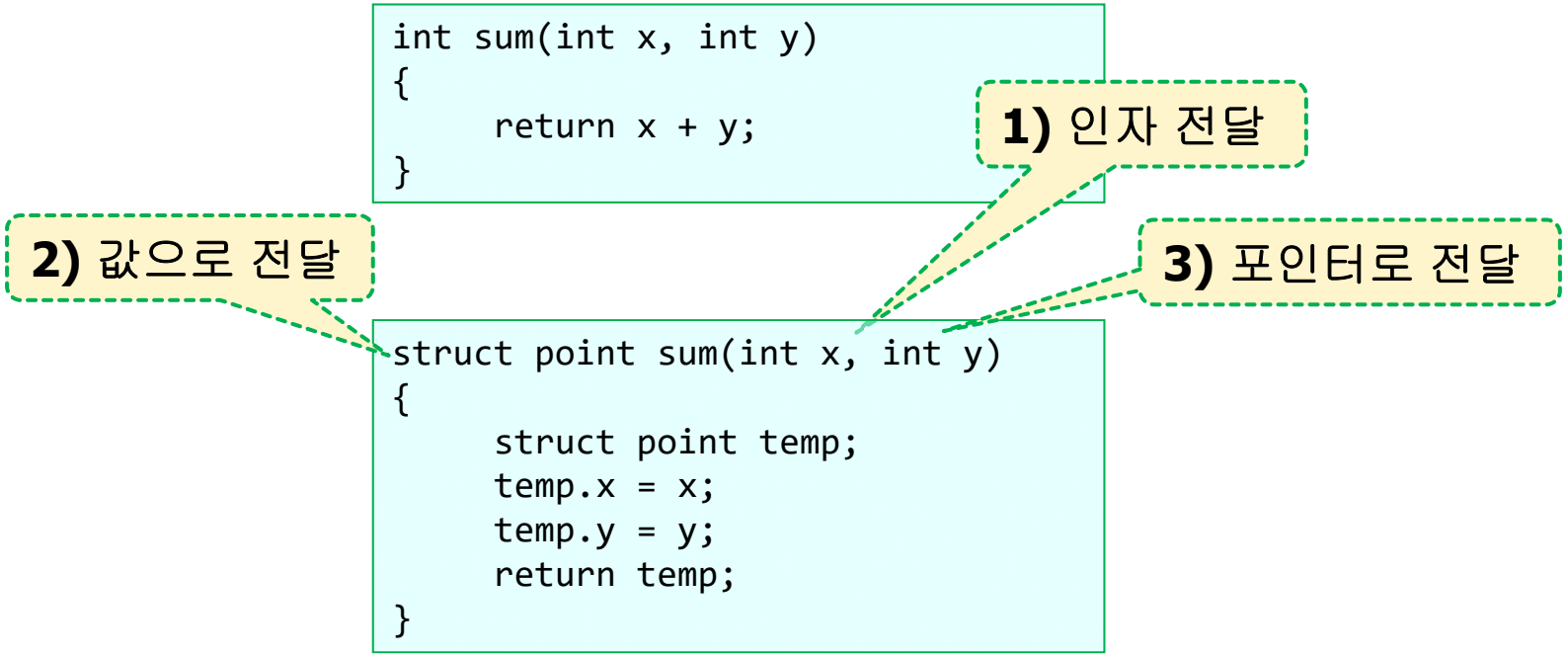
- 구조체의 중첩
  - ✓ 기존의 구조체를 활용한 새로운 구조체 정의

```
#include <stdio.h>
struct point {
        int x;
        int y;
};
struct rect {
        struct point pt1;
        struct point pt2;
};
int main(void)
{
        struct rect myscreen;
        int area;
        myscreen.pt1.x = 3;
        myscreen.pt1.y = 6;
        myscreen.pt2.x = 9;
        myscreen.pt2.y = 12;

        area = (myscreen.pt2.x - myscreen.pt1.x) *
               (myscreen.pt2.y - myscreen.pt1.y);
        printf("Area = %d\n", area);
        return 0;
}
```



구조체와 함수 (1/4)

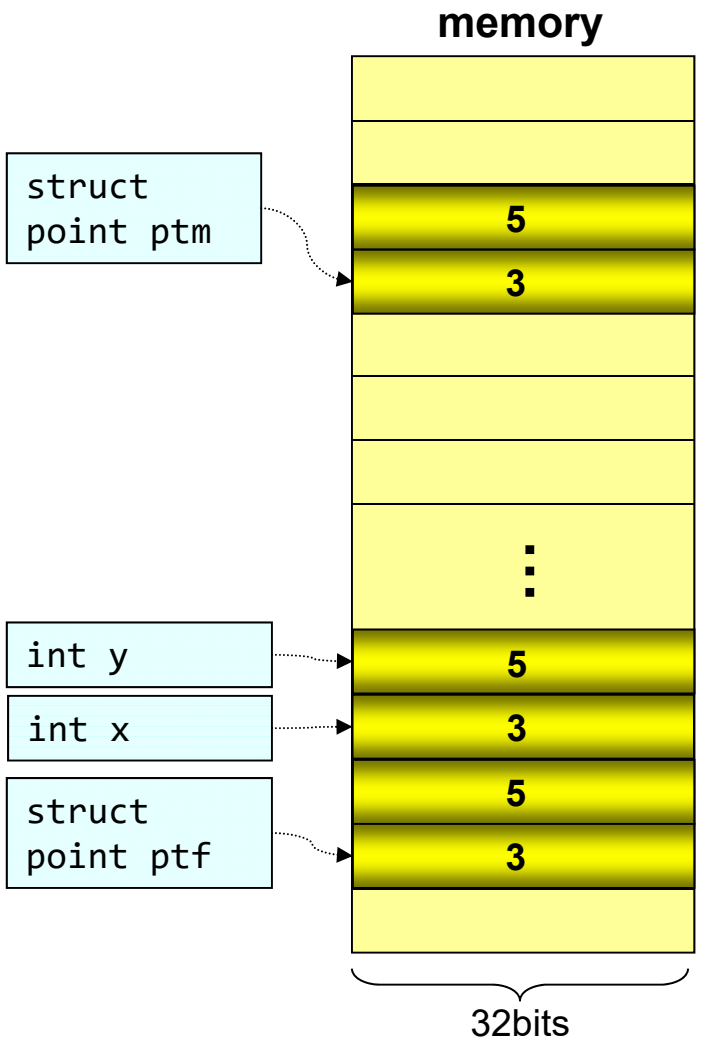
- 함수에서 구조체를 사용하는 방법
  - ✓ 1) 함수에 독립적인 인자로 전달
  - ✓ 2) 구조체를 값으로 전달
  - ✓ 3) 구조체를 포인터로 전달



구조체와 함수 (2/4)

- 1) 함수에 독립적인 인자로 전달

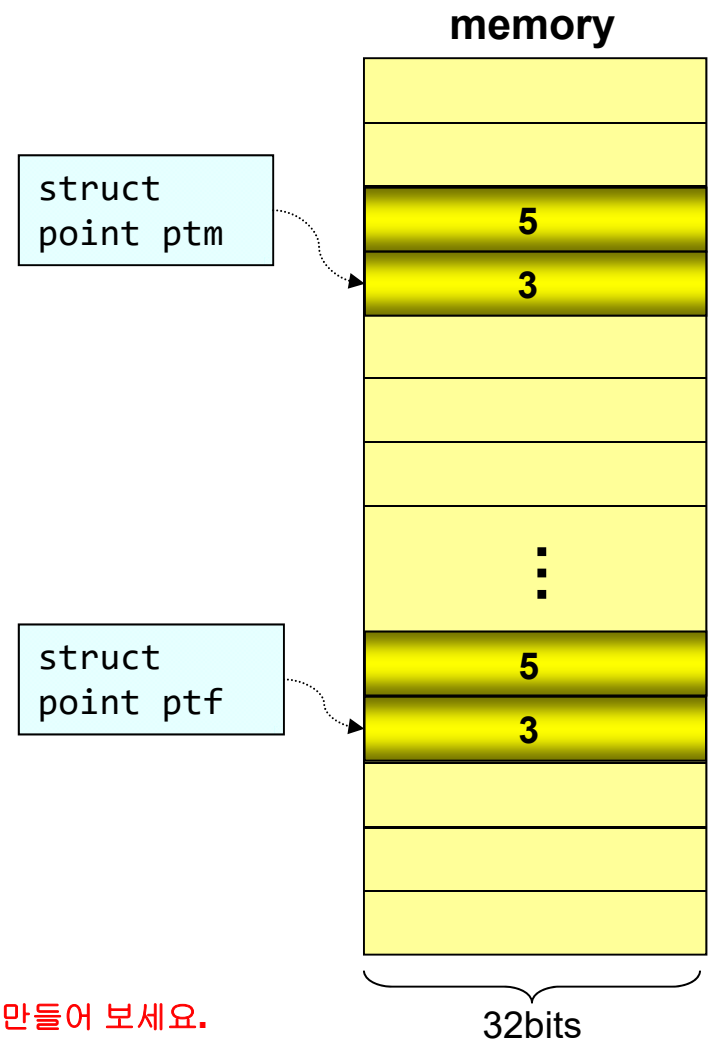
```
#include <stdio.h>

struct point {
    int x;
    int y;
};

void Print_Point(int x, int y)
{
    struct point ptf;
    ptf.x = x;
    ptf.y = y;
    printf("x, y = (%d, %d)\n", ptf.x, ptf.y);
}

int main()
{
    struct point ptm = {3, 5};

    Print_Point(ptm.x, ptm.y);

    return 0;
}
```



구조체와 함수 (3/4)

- 2) 구조체를 값으로 전달
  - ✓ Call by value

```
#include <stdio.h>

struct point {
    int x;
    int y;
};

void Print_Point(struct point ptf)
{
    printf("x, y = (%d, %d)\n", ptf.x, ptf.y);
}

int main()
{
    struct point ptm = {3, 5};

    Print_Point(ptm);

    return 0;
}
```

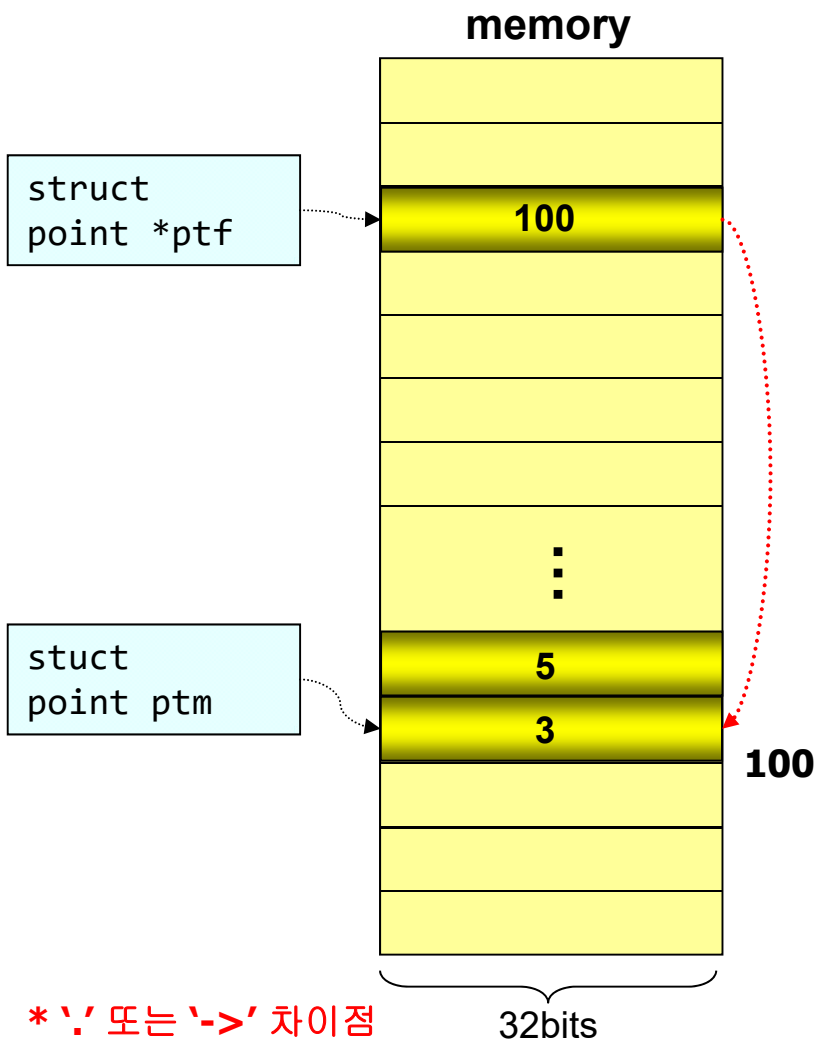


* **Print_Point()**를 수정하여 **Get_Point()** 로 만들고 리턴 값을 구조체로 만들어 보세요.
- 좌표 이동 **x, y** 값에 **5**를 더한다.

구조체와 함수 (4/4)

- 3) 구조체를 포인터로 전달
  - ✓ Call by reference

```
#include <stdio.h>

struct point {
    int x;
    int y;
};

void Print_Point(struct point *ptf)
{
   printf("x, y = (%d, %d)\n",
                          (*ptf).x, (*ptf).y);
   (*ptf).x += 1;
   (*ptf).y += 1;
}

int main()
{
    struct point ptm = {3, 5};

    Print_Point(&ptm);
    printf("x, y = (%d, %d)\n", ptm.x, ptm.y);
    return 0;
}
```



* '.' 또는 '->' 차이점

구조체와 포인터

- 포인터를 통한 구조체 접근

```
#include <stdio.h>
#include <string.h>
struct person{
        char name[20];
        int age;
        int height;
};
#define NR_TEAM 5
int main(void)
{
        struct person team[NR_TEAM];
        struct person *pptr;
        pptr = team;
        int i;
        for(i=0; i<NR_TEAM; i++){
                printf("Enter Nr=%d team member info(name age height): ", i);
                scanf("%s %d %d", (pptr+i)->name, &(pptr+i)->age, &(pptr+i)->height);
        }
        for(i=0; i<NR_TEAM; i++){
                printf("%s:%d:%d\n", (pptr+i)->name, (pptr+i)->age, (pptr+i)->height);
        }
        return 0;
}
```

구조체 typedef

- typedef를 통한 자료구조 정의

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

typedef int MYINT;
typedef char MYCHAR;

typedef struct person{
        MYCHAR name[20];
        MYINT age;
        MYINT height;
} PERSON;
int main(void)
{
        PERSON i1;
        strcpy(i1.name, "Testname");
        i1.age = 20;
        i1.height = 170;

        printf("%s:%d:%d\n", i1.name, i1.age, i1.height);

        struct person i2 = {"Test2name", 30, 180};
        printf("%s:%d:%d\n", i2.name, i2.age, i2.height);

        return 0;
}
```

구조체의 동적 할당

- malloc과 구조체 pointer 활용한 구조체의 동적 할당

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAXNAME 18
typedef struct person{
        char name[MAXNAME];
        int age;
        int height;
} PERSON;

int main(void)
{
        PERSON *new;
        printf("sizeof struct person=%d\n", sizeof(PERSON));

        new = (PERSON *)malloc(sizeof(PERSON));
        printf("Enter team member info(name age height): ");
        scanf("%s %d %d", (new)->name, &(new)->age, &(new)->height);

        printf("%s:%d:%d\n", new->name, new->age, new->height);
        return 0;
}
```

26?

회원 관리 프로그램(1/3)

- Using double linked list

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

#define MAXNAME 20
typedef struct person{
        char name[MAXNAME];
        int age;
        int height;
        struct person *prev;
        struct person *next;
} PERSON;

PERSON * head;

PERSON * GetNewNode(void) {
        PERSON * new = (PERSON *)malloc(sizeof(PERSON));
        new->prev = NULL;
        new->next = NULL;
        return new;
}
```

회원 관리 프로그램(2/3)

- Using double linked list

```
void InsertAtHead(PERSON * new) {
        if(head == NULL) {
                head = new;
                return;
        }
        head->prev = new;
        new->next = head;
        head = new;
}

void InsertAtTail(PERSON * new) {
        PERSON * temp = head;
        if(head == NULL) {
                head = new;
                return;
        }
        while(temp->next != NULL) temp = temp->next;
        temp->next = new;
        new->prev = temp;
}
```

회원 관리 프로그램(3/3)

- Using double linked list

```
void Print() {
        PERSON * curr= head;
        printf("Forward search\n");
        while(curr != NULL) {
                printf("%s:%d:%d\n", curr->name, curr->age, curr->height);
                curr = curr->next;
        }
}

int main(void)
{
        PERSON * curr;
        int i;
        head = NULL;

        for (i=0; i<10; i++) {
                curr = GetNewNode();
                printf("Enter Nr=%d team member info(name age height): ", i);
                scanf("%s %d %d", (curr)->name, &(curr)->age, &(curr)->height);
                InsertAtHead(curr);
        }
        Print();

        return 0;
}
```

과제

- 이름, 나이, 키를 저장할 수 있는 회원관리 프로그램 작성
  - ✓ Keyword: structure, dynamic allocation, double linked list
  - ✓ 메모리가 허락하는 한 무한대의 회원 추가 기능
    - 회원 추가 시 이름순(오름차순) 정렬하여 insert
  - ✓ 회원 삭제 기능
  - ✓ 회원 print 기능
  - ✓ 이름으로 검색 기능
  - ✓ Help 기능
  - ✓ Bonus: 이름/나이/키로 정렬 기능(동일한 key값인 경우 다른 key를 기준으로 하여 정렬)

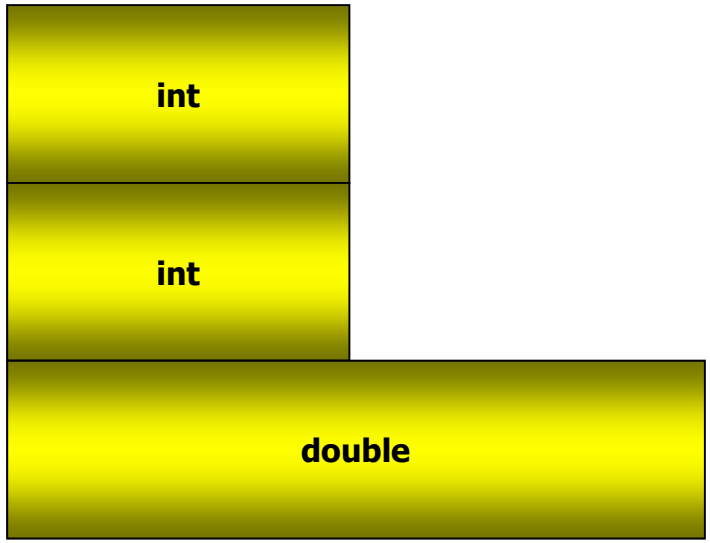
union

- Union: 다른 종류의 데이터를 하나의 저장 공간에서 다룰 수 있도록 지원

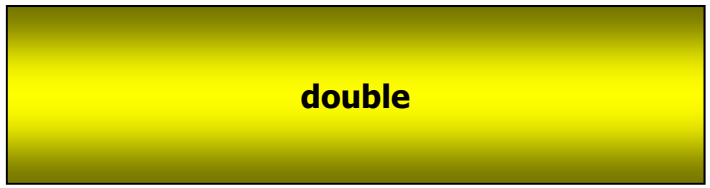
```
typedef struct sbox
{
      int mem1;
      int mem2;
      double mem3;
} SBox;
```

vs.

```
typedef union sbox
{
      int mem1;
      int mem2;
      double mem3;
} SBox;
```



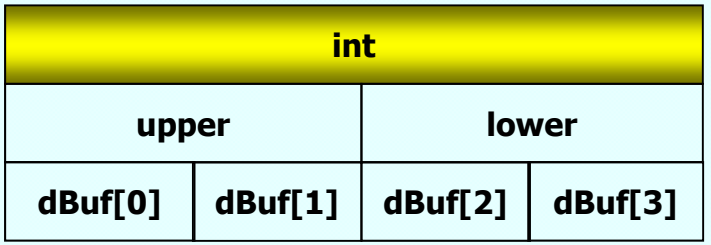
vs.



union

- union 키워드 사용

```
#include <stdio.h>
typedef struct dbshort
{
        unsigned short upper;
        unsigned short lower;
} DBShort;
typedef union rdbuf
{
        int iBuf;
        char bBuf[4];
        DBShort sBuf;
} RDBuf;
int main(void)
{
        RDBuf buf;
        printf("Enter an Interger: ");
        scanf("%d", &(buf.iBuf));
        printf("Upper 2B: %u\n", buf.sBuf.upper);
        printf("Lower 2B: %u\n", buf.sBuf.lower);
        printf("MSB [0] : %d\n", buf.bBuf[0]);
        printf("MSB [1] : %d\n", buf.bBuf[1]);
        printf("MSB [2] : %d\n", buf.bBuf[2]);
        printf("LSB [3] : %d\n", buf.bBuf[3]);
        return 0;
}
```



enum

- Enum: unique types with values ranging over a set of named constants called enumerators

```
#include <stdio.h>

enum month {JAN,FEB,MAR,APR,MAY,JUN,JUL,AUG,SEP,OCT,NOV,DEC};
//enum month {JAN=1,FEB,MAR,APR,MAY,JUN,JUL,AUG,SEP,OCT,NOV,DEC};
//enum month {JAN=10,FEB,MAR,APR,MAY,JUN,JUL,AUG,SEP,OCT,NOV,DEC};
//enum month {JAN=13,FEB,MAR,APR,MAY,JUN,JUL=83,AUG,SEP,OCT,NOV,DEC};

int main(void)
{
        int i;

        printf("%d\n", MAR);
        for(i=JAN;i<=DEC;i++)
                printf("%d ",i);

        return 0;
}
```

이 장의 결론

- 구조체의 개념을 이해
- 구조체와 함수
- 구조체의 배열과 포인터
- 구조체의 동적할당과 리스트
- Typedef, enum, union