Programming Methodology
Practice Session #3

Pointer, Call by Reference, and Structure

Pointer

- 메모리 주소 값을 저장하는 data type의 한 종류
  - **int** * p1과 같이 '*' 연산자를 이용하여 선언한다.
- & 연산자는 변수의 메모리 주소를 나타낸다.
- 포인터 변수의 이름 앞에 '*'을 붙여 사용하면 해당 포인터가 참조하고 있는 변수값을 얻을 수 있다.

```
int x = 5;
int * p = &x;              // x의 주소값을 p에 할당
printf("%d\n", p);         // x의 주소값 출력
printf("%d\n", *p);        // x의 값(5) 출력
```

Call by Reference (1)

- **Call by Value** (값에 의한 호출)
  - 함수를 호출할 때 parameter 값을 복사하여 사용한다.
  - 함수의 수행 도중 parameter의 값을 바꾸어도 호출 하기 전 원래의 변수 값은 바뀌지 않는다.

- **Call by Reference** (참조에 의한 호출)
  - 함수를 호출할 때 parameter의 주소를 복사하여 사용한다.
  - 함수의 수행 도중 parameter의 값을 바꾸면 호출 하기 전 원래의 변수 값도 바뀐다.
  - Pointer(*)를 이용한 call by pointer와 reference(&)를 이용한 call by reference로 분류하기도 한다.

Call by Reference (2)

```
void swapByValue(int a, int b)            // Call by Value
{   int temp = a;   a = b;   b = temp;   }

void swapByPointer(int * a, int * b)      // Call by Pointer
{   int temp = *a;   *a = *b;   *b = temp;   }

void swapByReference(int & a, int & b)    // Call by Reference
{   int temp = a;   a = b;   b = temp;   }

void main()
{
   int a = 3, b = 5;
   swapByValue(a, b);              // a = 3, b = 5 (바뀌지 않는다)
   swapByPointer(&a, &b);          // a = 5, b = 3 (바뀐다)
   swapByReference(a, b);          // a = 3, b = 5 (바뀐다)
}
```

Structure (1)

- 하나의 변수에 여러 가지의 데이터를 저장하기 위해 **struct** 키워드를 이용하여 structure를 만들 수 있다.

```
struct student {
    char name[20]; // 이름
    double GPA;    // 학점
};
struct student s1, s2;   // struct student변수 선언
```

- **typedef** 키워드를 사용하면 structure의 선언이 편리하다.

```
typedef struct _student {
    ...
} student;        // struct _student를 student로 선언 가능
student s1, s2;  // student 변수 선언
```

Structure (2)

- **struct** 변수 내의 field에 접근하기 위해 **'.'** 연산자를 사용한다.

```
struct student s1, s2;   // struct student변수 선언

printf("%s : %d\n", s1.name, s1.GPA);
printf("%s : %d\n", s2.name, s2.GPA);
```

- **struct** pointer 변수 내의 field에 접근하기 위해 **'->'** 연산자를 사용한다.

```
struct student *s1;
struct student *s2;      // struct student pointer 변수 선언

printf("%s : %d\n", s1->name, s1->GPA);
printf("%s : %d\n", s2->name, s2->GPA);
```

Sample Practice (1)

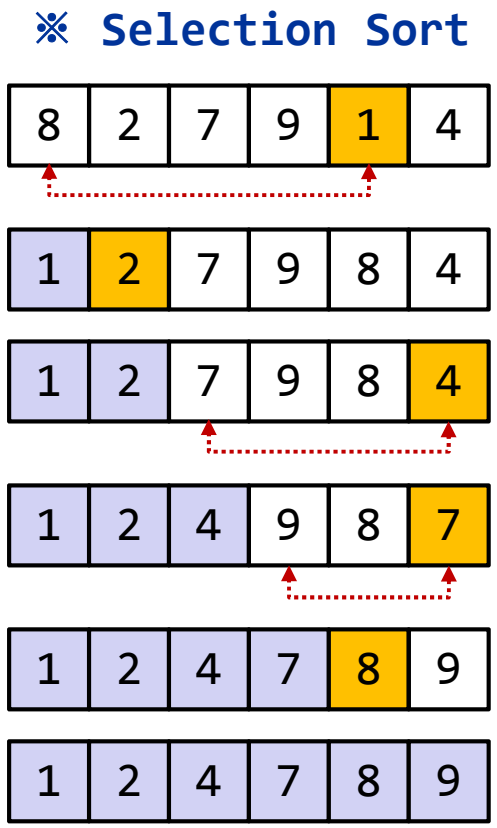
- **struct** Student 를 구현한다.

```
typedef struct Student {
    char name[20]; // 이름
    double GPA;    // 학점
} Student;
```

- Student 구조체 변수(call by reference)와 name, GPA를 parameter로 받아 구조체 변수에 name과 GPA를 저장하는 **setData(Student&, char*, double)** 함수를 구현한다.

Sample Practice (2)

- **struct** Student에 대한 **Selection Sort**를 구현한다.
  - name에 대해 **ascending order** (오름차순)으로 정렬하는 함수 sortByName()을 구현한다.
  - GPA에 대해 **descending order** (내림차순)으로 정렬하는 함수 sortByGPA()를 구현한다.
  - **void** sortByName (Student* list, **int** size);
  - **void** sortByGPA(Student* list, **int** size);



Sample Practice (3)

- `void swap(Student a, Student b)`
  - 위 prototype을 적절한 형태로 바꾸어 **call by reference** (call by pointer가 아닌) 로 구현한다.
  - Selection sort 과정에서 두 element를 바꿀 때 반드시 `swap()` 함수를 사용한다.

- `void printList(Student* list, int size)`
  - `Student` array를 출력하는 함수이다. 출력되는 숫자들의 열을 맞추기 위해 `"%10s : %.1f"`의 format으로 `name`과 `GPA`를 출력한다.

Sample Practice (4)

```
void main()
{
    Student list[10];

    setData(list[0], "John", 3.2);
    setData(list[1], "Christina", 2.8);
    setData(list[2], "Kate", 1.4);
    setData(list[3], "Michael", 4.0);
    setData(list[4], "George", 1.9);
    setData(list[5], "Bob", 2.4);
    setData(list[6], "James", 3.1);
    setData(list[7], "Katherine", 2.5);
    setData(list[8], "Eric", 0.4);
    setData(list[9], "Jessica", 3.4);

    printf("** Initial List **₩n");
    printList(list, 10);

    printf("₩n** Sorted by Name **₩n");
    sortByName(list, 10);
    printList(list, 10);

    printf("₩n** Sorted by GPA **₩n");
    sortByGPA(list, 10);
    printList(list, 10);
}
```

Sample Practice (5)

```
** Initial List **
      John : 3.2
 Christina : 2.8
      Kate : 1.4
   Michael : 4.0
    George : 1.9
       Bob : 2.4
     James : 3.1
 Katherine : 2.5
      Eric : 0.4
   Jessica : 3.4

** Sorted by Name **
       Bob : 2.4
 Christina : 2.8
      Eric : 0.4
    George : 1.9
     James : 3.1
   Jessica : 3.4
      John : 3.2
      Kate : 1.4
 Katherine : 2.5
   Michael : 4.0
```

```
** Sorted by GPA **
   Michael : 4.0
   Jessica : 3.4
      John : 3.2
     James : 3.1
 Christina : 2.8
 Katherine : 2.5
       Bob : 2.4
    George : 1.9
      Kate : 1.4
      Eric : 0.4
계속하려면 아무 키나 누르십시오 . . .
```