

정렬(quick)

PREP(quickSort를 위한 partition함수)

quickSort를 프로그래밍하기 이전에 그 안에서 사용되는 partition함수를 정의하여 테스트 하려한다.

0, 20, 55, 66, 10, 40, 88, 77, 30, 49의 원소들을 갖는 list배열을 partition(list, 0, 9)를 호출하였을 때의 수행 과정을 손으로 그려보라.
(수업시간에 한 대로)

```
#include <stdio.h>

int partition(int A[], int p, int r)
{
    // 배열 A[p..r]의 원소들을 A[r]을 기준으로
    // 양쪽으로 재배치하고
    // A[r]이 자리한 위치를 return한다.
}
```

```
void printList(int A[], int s, int e)
{
    int i;
    for (i = s; i <= e; i++)
        printf("%d ", A[i]);
    printf("\n");
}
```

```
int main(void)
{
    int list[] = {0, 20, 55, 66, 10, 40, 88, 77, 30, 49};
    int list2[] = {0, 20, 10, 40, 30, 49, 88, 77, 55, 66};
    int loc;

    // test 1
    printList(list, 0, 9);
    loc = partition(list, 0, 9); // list
    printList(list, 0, 9); // 결과는 list2의 값과 같게 된다.
    printf("%d의 위치는 %d\n", 49, loc); // 49의 위치는 5

    // test 2
    printList(list2, 6, 9); // 88, 77, 55, 66
    loc = partition(list2, 6, 9);
    printList(list2, 6, 9); // 55, 66, 88, 77
    printf("%d의 위치는 %d\n", 66, loc); // 66의 위치는 7

    // test 3
    printList(list2, 0, 4); // 0, 20, 10, 40, 30
    loc = partition(list2, 0, 4);
    printList(list2, 0, 4); // 0, 20, 10, 30, 40
    printf("%d의 위치는 %d\n", 30, loc); // 30의 위치는 3
}
```

Lab(quickSort)

- 이제는.. Quick sort로
 - 동적 할당 : n 개의 정수를 저장할 수 있는 배열 생성
 - 배열에는 랜덤 정수를 저장
 - quick sort로 정렬.
 - 프로그램 종료 전에 배열 반환

HW(quickSort)

- 학생의 성적을 정렬하는 프로그램 작성.
 - 임의의 학생의 수 n 을 입력 받음.
 - 학생은 학번, 영어, 수학, 국어 성적을 `attribute(member)`로 가진다 ➔ `struct`
 - 모든 `attribute`는 `int` 형
 - 학번은 1번부터 시작. 성적은 0~100점 사이의 랜덤 정수.
 - 국어 성적 기준으로 내림 차순으로 정렬
 - Quick sort로 정렬 (함수)

```
struct student
```

```
{
```

```
    int id; //학번. 1번 부터 부여 ..
```

```
    int korean, english, math;
```

```
};
```

```
int main()
```

```
{
```

```
    //n 입력 받음
```

```
    //구조체 배열을 동적으로 할당
```

```
    // 학번 부여
```

```
    // random으로 성적 저장
```

```
    // 학생 정보(학번, 성적들) 출력
```

```
    //국어 성적 기준으로 내림차순 정렬 → quick sort 함수 호출
```

```
    // 정렬된 학생 정보(학번, 성적들) 출력
```

```
    //동적으로 할당 받은 구조체 배열 반환
```

```
};
```

HW(quickSort2)

원하는 기준(국어/영어/수학/id)를 main에서 선택 후 그 기준에 따라 정렬하는 버전으로 작성하라.

- 하나의 quickSort와 하나의 partition 함수를 사용
- 그 외 다른 함수(swap?)
- Quick sort 사용
- Id는 오름차순으로, 성적으로 정렬 시는 내림차순
- 원할 때까지 반복적으로 기준을 선택해 정렬한다.

```
C:\windows\system32\cmd.exe
학생 수를 입력하세요 : 5
학번 : 1      국어 : 25      수학 : 78      영어 : 70
학번 : 2      국어 : 80      수학 : 15      영어 : 49
학번 : 3      국어 : 6       수학 : 53      영어 : 75
학번 : 4      국어 : 59      수학 : 74      영어 : 60
학번 : 5      국어 : 33      수학 : 19      영어 : 24

정렬 기준 선택<1: 국어, 2: 수학, 3: 영어, 4: id<오름차순>, 0: 끝내기> : 1
학번 : 3      국어 : 6       수학 : 53      영어 : 75
학번 : 1      국어 : 25      수학 : 78      영어 : 70
학번 : 4      국어 : 59      수학 : 74      영어 : 60
학번 : 2      국어 : 80      수학 : 15      영어 : 49
학번 : 5      국어 : 33      수학 : 19      영어 : 24

정렬 기준 선택<1: 국어, 2: 수학, 3: 영어, 4: id<오름차순>, 0: 끝내기> : 2
학번 : 1      국어 : 25      수학 : 78      영어 : 70
학번 : 4      국어 : 59      수학 : 74      영어 : 60
학번 : 3      국어 : 6       수학 : 53      영어 : 75
학번 : 5      국어 : 33      수학 : 19      영어 : 24
학번 : 2      국어 : 80      수학 : 15      영어 : 49

정렬 기준 선택<1: 국어, 2: 수학, 3: 영어, 4: id<오름차순>, 0: 끝내기> : 3
학번 : 2      국어 : 80      수학 : 15      영어 : 49
학번 : 4      국어 : 59      수학 : 74      영어 : 60
학번 : 5      국어 : 33      수학 : 19      영어 : 24
학번 : 1      국어 : 25      수학 : 78      영어 : 70
학번 : 3      국어 : 6       수학 : 53      영어 : 75

정렬 기준 선택<1: 국어, 2: 수학, 3: 영어, 4: id<오름차순>, 0: 끝내기> : 4
학번 : 1      국어 : 25      수학 : 78      영어 : 70
학번 : 2      국어 : 80      수학 : 15      영어 : 49
학번 : 3      국어 : 6       수학 : 53      영어 : 75
학번 : 4      국어 : 59      수학 : 74      영어 : 60
학번 : 5      국어 : 33      수학 : 19      영어 : 24

정렬 기준 선택<1: 국어, 2: 수학, 3: 영어, 4: id<오름차순>, 0: 끝내기> : 0
계속하려면 아무 키나 누르십시오 . . .
```