Hybrid Quick Sort
[code] DS1901-HW03

Problem

Partition을 한 후 subarray의 원소의 개수가 t개 이하일 경우 재귀호출하지 않고 insertion sort를 수행하는 hybridsort() 함수를 구현하시오. 이 때 return value는 insertion sort를 호출한 횟수이다.

[제약조건]

- 함수의 prototype은 다음과 같다.
  int hybridsort(int A[], int p, int q, int t);
  p, r : 배열 A에서의 인덱스
  t : 원소의 개수가 t개 이하이면 insertion sort를 호출하는 임계값
  반환값 : A[p..q]를 정렬하는 동안 insertion sort를 호출한 횟수
- 반드시 실습시간에 구현한 partition 함수를 사용할 것 (randomize version을 사용하면 안 됨)
- 주의) **제출하는 소스파일에는 main 함수는 포함하지 않는다.**
  (다음 페이지 예처럼 mycode.c만 제출한다.)

[Hints]

- 실습 시간 구현한 quick sort, insertion sort 코드를 변경하면 된다.

Submission

Due: 4월 19일 (금) 23시 59분 59초

19950001@ubuntu:~/DS$ submit DS1901-HW03 mycode.c

Example of main()

```
///// main.c
#include <stdio.h>
#include <stdlib.h>

int hybridsort(int A[], int p, int q, int t);
int main(int argc, char *argv[]) {
int n=9, t=3;
      int A[9] = {1, 8, 6, 3, 2, 7, 4, 9, 10};
      for ( int i=0; i<9; i++ ) printf("%d ", A[i]);
      printf("\n");
      int res=hybridsort(A, 0, n-1, t);
      printf("No of calls = %d\n", res);
      for ( int i=0; i<9; i++ ) printf("%d ", A[i]);
      printf("\n");
      return 0;
}
```

```
///// mycode.c
…
int hybridsort(int A[], int p, int r, int t) {
            …
            …
}
…
```

[2개의 소스파일로 구현하여 테스트하는 방법]

```
$ gcc main.c mycode.c
$ ./a.out
1 8 6 3 2 7 4 9 10
No of calls = 4
1 2 3 4 6 7 8 9 10
```

Self Test

```
19950001@ubuntu:~/DS$ sftest DS1901-HW03 mycode.c
MSG> hw03.c was compiled.
1 th. Testing 100 5 ---
Your answer is 29 : success.

2 th. Testing 100 10 ---
Your answer is 16 : success.

3 th. Testing 200 5 ---
Your answer is 65 : success.
```