Lab 11 + Homework 3

실습문제는 두 명씩 짝을 지어 상의하면서 푼다.

숙제는 문제별로 한 파일에 작성하고 zip 으로 묶어서 제출사이트를 통하여 개별 제출합니다.
제출 마감은 12월 11일(일) 자정 1분전 (11:59 p.m.)입니다.
모범답은 마감 직후에 공개합니다. 마감연장은 없습니다.
문제에 대해서 학우들과 토론을 할 수는 있지만, 작성하는 코드는 자신이 직접 작성한 코드를 제출해야 합니다. 베껴서 제출한 코드로 판명되면 원본/복사본 모두 0점 처리합니다.

실습문제 1

`data` 라는 텍스트 파일에 학생 이름, 학번, 성적이 다음과 같은 형식으로 적혀있다.

```
Casanova  2016321368  A
Turing    2016332437  C
Hanyang   2016101213  B
. . .
```

성적은 `A`, `B`, `C`, `D`, `F`의 다섯 등급으로 나누어진다고 하고, 이 파일의 데이터를 읽어서 `struct student` 타입의 배열 `class` 배열에 저장하는 프로그램을 `reorder.c` 라는 이름의 파일에 작성하자.
프로그램에서 `data` 파일에 있는 텍스트를 읽으려면, 데이터 파일을 프로그램 파일과 같은 폴더에 두고, 커맨드 창에서 프로그램을 다음과 같이 실행 시킨다.

```
$ gcc reorder.c –o reorder
$ reorder < data
```

읽은 데이터는 성적순으로 정렬하여 창에 프린트한다. 같은 성적의 학생은 알파벳 순으로 정렬하여 프린트 한다.
그리고 마지막 줄에는 학급의 성적 평균을 계산하여 프린트 한다. 평균을 계산하는 class_average() 함수를 작성하여 이 작업을 수행하도록 한다. A 학점은 평점 4, B 학점은 평점 3, C 학점은 평점 2, D 학점은 평점 1, F 학점은 평점 0으로 계산한다.

숙제 1

강의 시간에 배운 이분검색나무(binary search tree)를 마디를 특정 순서로 마디를 방문하여 마디에 있는 단어를 알파벳순으로 배열 key에 저장하는 함수를 다음 프로토타입에 맞추어 작성하자.

```
void inorder_traversal(struct tnode *p, char *key[]);
```

힌트 : 이분검색나무의 마디를 다음 순서로 방문하면서 단어를 가져와서 배열에 저장한다.

1. 왼아래나무에 있는 단어를 모두 알파벳 순으로 배열에 저장한다.
2. 뿌리마디의 단어를 저장한다.
3. 오른아래나무에 있는 단어를 모두 알파벳 순으로 배열에 저장한다.

숙제 2

이분검색나무에서 키워드의 빈도수를 줄이는 함수 reducetree를 다음 함수 프로토타입에 맞추어 작성하자.

```
struct tnode *reducetree(struct tnode *p, char *w);
```

만약 단어 w가 이분검색나무 p에 있고 빈도수가 2 이상이면 빈도수를 1 감소시키고, 빈도수가 1 이면 그 마디를 나무에서 제거한다. 이 때, 그 제거 하는 마디의 하위에 있는 마디는 제 위치를 찾아주어 이분검색나무의 성질을 그대로 유지하고 있어야 한다. 만약 단어 w가 이분검색나무 p에 없으면, 수정없이 그대로 내준다.