
# 430.217
# Introduction to Data Structures

## Assignment 1. Lists

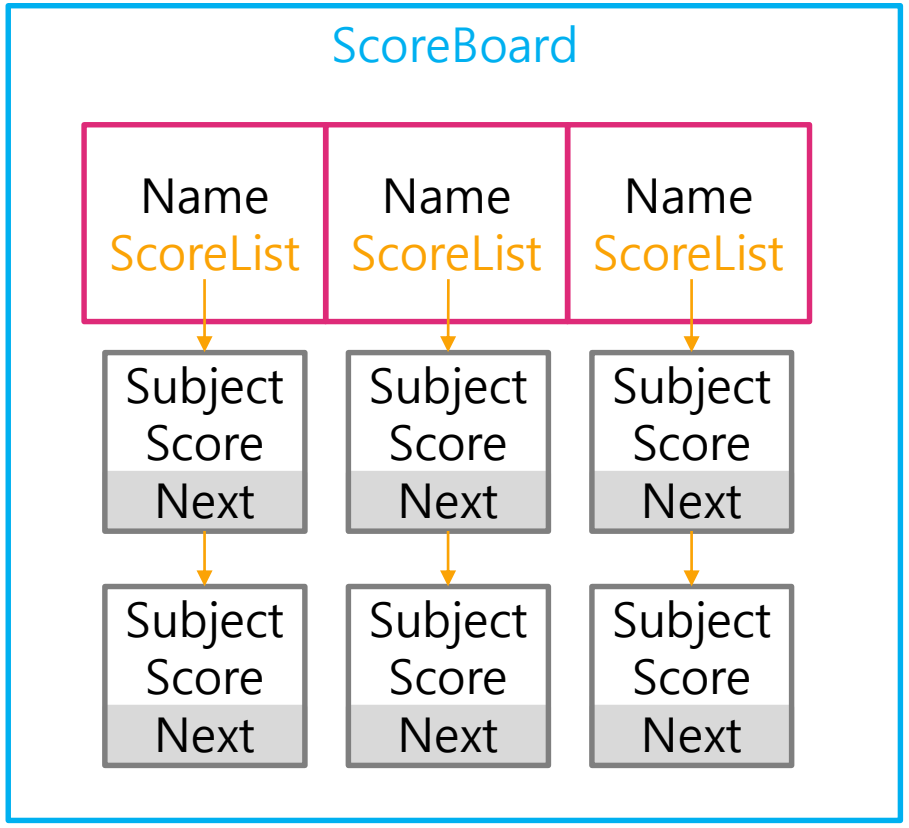
Seoul National University
Advanced Computing Laboratory

ACL ADVANCED COMPUTING LABORATORY

# 성적 관리 프로그램

- 한 교실 학생들의 성적을 관리하는 데이터 구조체
  - Class ScoreBoard
  - Class Student
  - Class LinkedList
  - Class Node

    (LinkedList와 Node는 그대로 사용)



# ScoreBoard

- Class declaration

```
class ScoreBoard
{
public:
    ScoreBoard(int n);
    ~ScoreBoard();

    void set_subjects(string* subjectArray, int numOfSubjects);
    void insert_score(string name, string subjectName, int score);

    int get_max_score_of_subject(string subjectName);
    Node* get_max_score_of_student(string name);
    string get_top_student_in_subject(string subjectName);
    string get_top_student();
    float get_average_of_subject(string subjectName);
    float get_average_of_student(string name);
    float get_average_of_class();

    void transfer(string name);
    void abolish_subject(string subjectName);
    void print();
private:
    string* subjectArray;
    int numOfSubjects;
    Student* studentArray;
    int numOfStudents;
};
```

# ScoreBoard – member variables

- 과목

```
string* subjectArray;
int numOfSubjects;
```

  - 과목 목록은 set_subjects 라는 member function으로 지정될 것
  - 그에 따라 과목 수도 결정

- 학생들의 데이터

```
Student* studentArray;
int numOfStudents;
```

  - Constructor에서 학생 수 입력 받음
  - 그에 따라 dynamic allocation 할 것

# ScoreBoard – member functions

- Constructor

```
ScoreBoard(int n);
```

  - 학생 수를 parameter로 받아, 이 크기만큼 student array 생성 및 학생 수 지정

- Destructor

```
~ScoreBoard();
```

  - Memory release

# ScoreBoard – member functions

- Set_subjects

```
void set_subjects(string* subjectArray, int numOfSubjects);
```

  - Member variable 지정

- Insert_score

```
void insert_score(string name, string subjectName, int score);
```

  - 해당 학생의 score list에 과목과 점수 데이터 입력

# ScoreBoard – member functions

- Get_max_score_of_subject

```
int get_max_score_of_subject(string subjectName);
```

  - 해당 과목의 최고점을 찾아내어 return

- Get_max_score_of_student

```
Node* get_max_score_of_student(string name);
```

  - 해당 학생의 score list 중 최고점을 찾아 해당 과목과 점수 return

# ScoreBoard – member functions

- Get_top_student_in_subject

```
string get_top_student_in_subject(string subjectName);
```

  - 해당 과목의 최고점을 가진 학생의 이름 return

- Get_top_student

```
string get_top_student();
```

  - 평균점수가 가장 높은 학생의 이름 return

# ScoreBoard – member functions

- Get_average_of_

```
float get_average_of_subject(string subjectName);
float get_average_of_student(string name);
float get_average_of_class();
```

  - Subject: 해당 과목의 평균을 구하여 return
  - Student: 해당 student의 평균을 구하여 return
  - Class: 이 학급 전체의 평균을 구하여 return

# ScoreBoard – member functions

- Transfer

```
void transfer(string name);
```

  - 해당 학생이 전학을 간 경우로, student array에서 없애기 (reallocation)
  - 학생 수 update

- Abolish_subject

```
void abolish_subject(string subjectName);
```

  - 해당 과목이 폐지된 것으로, subject array에서 없애기 (reallocation)
  - 과목 수 update
  - 각 학생의 해당 과목 score 데이터도 삭제

# ScoreBoard – member functions

- Print

```
void print();
```

  - 현재 ScoreList 현황 출력
  - 예시

studentArray

| John LinkedList | Tom LinkedList | Bob LinkedList |
|---|---|---|

subjectArray

| Math | Physics | Biology |
|---|---|---|

표현방법:

| ₩t | Math₩t | Physics₩t | Biology₩t | Average₩t |
|---|---|---|---|---|
| John₩t | 92₩t | 87₩t | 79₩t | 86.0000₩t |
| Tom₩t | 85₩t | 88₩t | 81₩t | 84.6667₩t |
| Bob₩t | 80₩t | 79₩t | 85₩t | 81.3333₩t |

화면출력:

```
        Math    Physics Biology Average
John    92      87      79      86.000000
Tom     85      88      81      84.666667
Bob     80      79      85      81.333333
```

# Student

- Class declaration

```
class Student
{
public:
    string name;
    LinkedList* scoreList;

    Student();
    ~Student();
};
```

- 학생 이름과 이 학생의 score list 정보를 저장

- Constructor
  - Memory allocation

- Destructor
  - Memory release

# 주의사항

- Header file(.h)은 주어졌으니, 그에 맞는 .cpp 파일을 구현
  - Ex) LinkedList.h 에서 선언한 것을 LinkedList.cpp에서 구현

- LinkedList (각 학생의 score list) 의 실질적 정보는 private이므로,
  정보에 접근하기 위해서는 locate 함수를 잘 이용할 것

- 제출
  - ETL에서 assignment 1 에 제출할 것
  - Due: 10/4(일) 23:59
  - 위 assignment를 구현한 Visual studio 프로젝트 (내문서 안에 존재)
    폴더 자체를 압축하여 제출
  - 압축파일 이름 형식: 2015-12345 (.zip)