Data Structures

Lab # 01

Data & Knowledge Engineering Laboratory
Kyung Hee University, Korea

Chapter 2 Programming Exercise

실습에 사용할 샘플 코드 소개

- **교재의 Case Study에 대한 소스 코드는 교재 출판사에서 제공됨**
  - http://www.jblearning.com/ 에서 C++ plus data structures로 검색 후 Sample Materials 탭에서 다운로드
  - 복사본을 KLAS에서 제공
- **앞으로 모든 실습에 사용할 샘플 코드**
  - 실습은 이 소스코드를 사용함을 가정하고 진행될 예정

CR/LF 설명

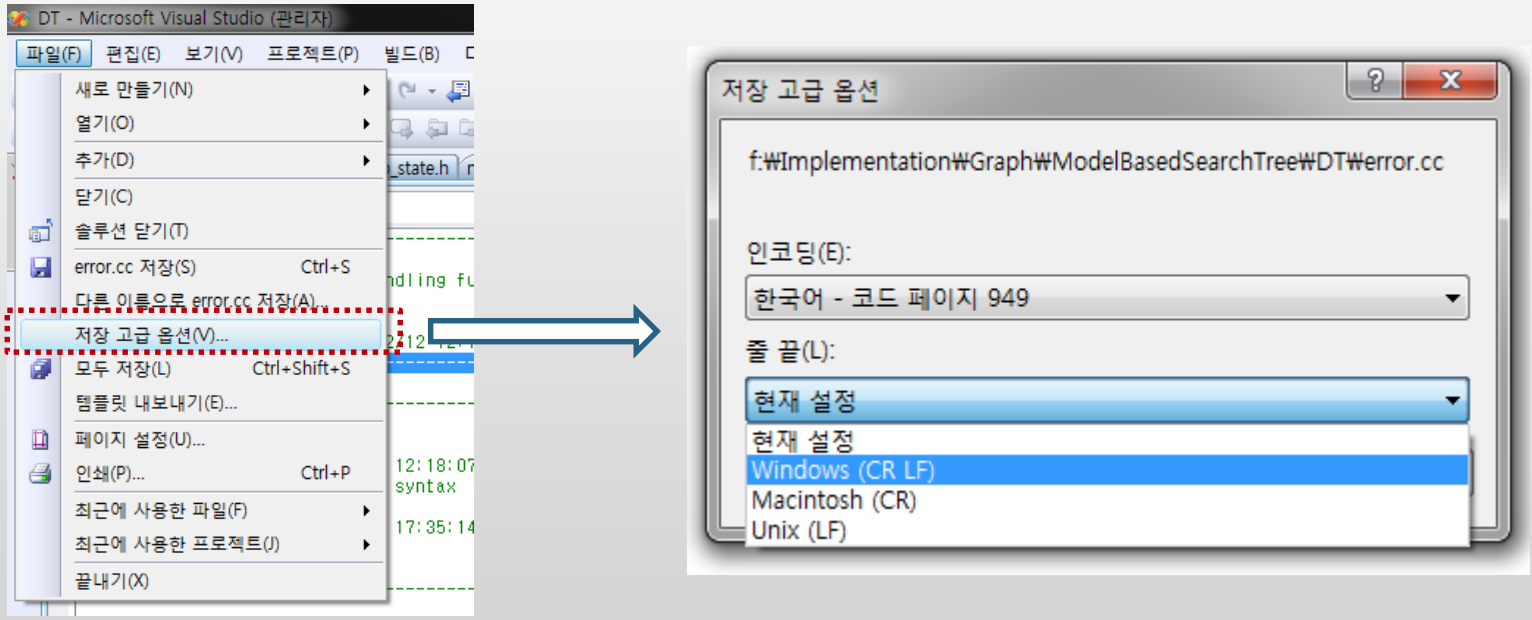
CR/LF

- CR : Carriage Return -> (Macintosh)
- LF : Line Feed -> (Unix)
- CR LF -> (Windows)

소스 코드 변환

■ 원본 소스의 개행문자를 반드시 변환해야 함

- ❖ MAC에서 작성한 소스코드 이기 때문에 개행문자가 LF 로 되어있음
- ❖ 윈도우에서는 개행문자를 CR/LF로 사용하기 때문에 원본 소스를 사용할 수 없으므로 변환이 필요함
- ❖ VS 2008에서는 아래와 같은 방법으로 변환 할 수 있음



Lab 1 전체 과제

1. Exercise 14 (한글 교재 14)
2. Exercise 15 (한글 교재 15)
3. Exercise 16 (한글 교재 16)
4. Exercise 28 (한글 교재 23)
5. Exercise 29 (한글 교재 24)

1. Exercise 14 (한글 교재 14)

■ 문 제

- ❖ StudentRecord에 대하여 멤버 길이 오프셋(offset) 테이블을 만들어라
- ❖ 정수 자료형(int)의 크기는 4byte, 실수 자료형(float)의 크기는 4byte로 계산

```
typedef char String[9];
struct StudentRecord
{
   String firstName;
   String lastName;
   int id;
   float gpa;
   // char gender;
   int currentHours;
   int totalHours;
};
StudentRecord student;
StudentRecord students[100];
```

선언된 StudentRecord 구조체

	Length	Offset
firstName	?	?
lastName	?	?
Id	?	?
gpa	?	?
currentHours	?	?
totalHours	?	?

[Offset Table]

■ 제출 방법

- ❖ 한글 문서에 offset 테이블을 작성

1-help slide

Offset (변위)

- 현재 주소의 위치를 찾기 위하여 기준이 되는 주소에 더해지는 값
- 자료형, 객체, 배열에 할당된 메모리 크기를 알기 위해서, C에서 제공하는 sizeof (type) 을 이용하여 현재 변수의 byte수를 측정할 것
  - Ex) sizeof(int) = 4

Offset Table

- Length는 구조체 내의 변수의 길이를 의미함

2. Exercise 15 (한글 교재 15)

문 제

❖ 만약 student의 기본 주소가 100이면, student.gpa의 주소는 무엇인가?
student.gpa = 3.87;

```
typedef char String[9];
struct StudentRecord
{
   String firstName;
   String lastName;
   int id;
   float gpa;
   // char gender;
   int currentHours;
   int totalHours;
};
StudentRecord student;
StudentRecord students[100];
```

선언된 StudentRecord 구조체

Address(student.gpa) = Base_Address(student) + Offset(gpa)

제출 방법

❖ 한글 문서에 계산하는 과정과 결과를 같이 작성할 것

3. Exercise 16 (한글 교재 16)

■ 문 제

❖ students 에 대하여 컴파일러는 얼마나 많은 메모리 공간을 준비하는가?

```
typedef char String[9];
struct StudentRecord
{
   String firstName;
   String lastName;
   int id;
   float gpa;
   // char gender;
   int currentHours;
   int totalHours;
};
StudentRecord student;
StudentRecord students[100];
```

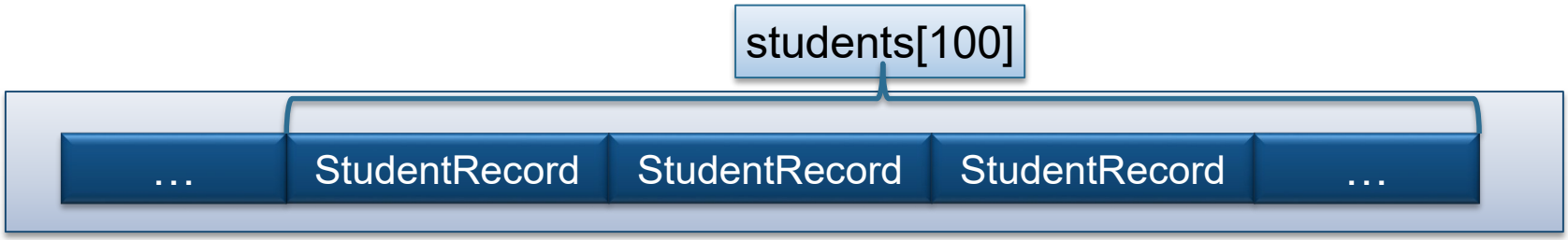
선언된 StudentRecord 구조체

■ 제출 방법

❖ 한글 문서에 계산하는 과정과 결과를 같이 작성할 것

3-help slide

- **Students 는 StudentRecord 구조체의 연속적인 배열임**



메모리 상에 잡힌 students

- **sizeof (type) 를 이용하여 StudentRecord 객체의 크기(bytes)를 측정함**
  - ❖ sizeof(StudentRecord) : StudentRecord객체 하나의 바이트 수를 가져올 수 있음

4. Exercise 28 (한글 교재 23)

■ 문 제

- ❖ a. ADT SquareMatrix에 대한 규격 명세를 작성하라. (정사각형 행렬(square matrix)은 N행과 N열을 가진 2차원 배열에 의해 표현될 수 있다.) 최대 크기로 50개의 행과 열을 가정할 수도 있다. 다음의 연산들을 포함하여라.

> MakeEmpty(n) : 모든 n행과 열들을 0으로 설정한다.
> StoreValue(i, j, value) : value를 i번째행, j번째 열의 위치에 저장한다.
> Add : 두 행렬을 함께 더한다.
> Subtract : 한 행렬을 다른 행렬로부터 뺀다.
> Copy : 한 행렬을 다른 행렬로 복사한다.

- ❖ b. 위의 규격 명세를 c++ 클래스 선언으로 변화하라.
- ❖ c. 멤버 함수들을 구현하라

■ 제출 방법

- ❖ 소스코드 내에 주석으로 규격 명세를 작성하고, 클래스를 직접 구현하여 소스코드를 제출할 것

4-help slide (1/3)

- **A) 다음을 참고하여 ADT 명세서를 작성하세요.**

SquareMatrix ADT Specification

구조	
	?

연산

MakeEmpty(n)	
기능: 조건: 결과:	Matrix의 n 행 열 내부를 0으로 초기화 N의 최대 크기는 50 N 안의 행 열은 0으로 초기화

StoreValue(i, j, value)	
기능: 조건: 결과:	?

4-help slide (2/3)

- **B) C++ 문법으로 클래스를 작성하세요.**

```
//#define MAX_ROWS 50
//#define MAX_COLS 50
const int MAX_ROWS = 50;
class SquareMatrix
{
private:
  int matrix[50][50];
  …
public:
  void MakeEmpty(int);
  void StoreValue(int, int, int);
  …

};
```

SquareMatrix 클래스 선언. 나머지 부분을 채워 넣으세요

4-help slide (3/3)

- C) 클래스에 정의한 멤버 함수를 각각 구현하세요.

```
void SquareMatrix::MakeEmpty(int n)
{
  int i, j = 0;
  for(i = 0; i <n; i++)
    for(j = 0; j < n; j++)
      matrix[i][j] = 0;
}
```

```
void SquareMatrix::StoreValue(int i, int j, int value)
{
  matrix[i][j] = value;
}
```

5. Exercise 29 (한글 교재 24)

■ 문 제

❖ 다음의 규격 명세의 ComparedTo 함수를 추가하여 StrType을 향상시켜라.

24. 다음의 규격 명세의 ComparedTo 함수를 추가하여 StrType을 향상시켜라.

```
RelationType ComparedTo(StrType& otherString)
기능:        self와 otherString을 알파벳 순으로 비교한다.
조건:        self와 otherString은 초기화되었다.
결과:        함수 값 = LESS, 만약 self가 otherString 앞에 온다면
             = GREATER, 만약 self가 otherString 뒤에 온다면
             = EQUAL, 만약 self가 otherString 과 같다면
```

a. <cstring>에 있는 strcmp를 사용하여 멤버함수 ComparedTo를 작성하라.
b. strcmp 라이브러리 함수를 사용하지 않고, 멤버 함수 ComparedTo를 다시 작성하라.

StrType 소스파일
- 다운 받은 chapter2 폴더의 String test 폴더 참조
- 경로 : labplus₩lab,c++ 3rd ₩chapter2₩String test₩

■ 제출 방법

❖ 소스코드 제출

5-help slide (1/2)

A) strcmp를 사용하여 함수 작성

- 헤더파일 : cstring.h
- int strcmp(const char *s1, const char *s2);
  - 두 문자열 s1과 s2를 비교함,
    - If s1 < s2 then return -1, Else if s1 == s2 then return 0, Else return 1
  - If (!strcmp(s1,s2))로 사용하는것보다 if(strcmp(s1,s2)==0) 으로 사용하는것이 더 바람직함, 컴파일러에 따라서 논리 에러가 발생할 수 있음
  - Example)

```
int main()
{
    char buf[80];
    while (fgets(buf, 80, stdin) != NULL)
    {
        buf[strlen(buf)-1] = 0x00;
        if(strcmp(buf, "exit") == 0)
            exit(0);
    }
}
```

- int strlen(char* str);
  - 문자열내의 문자의 개수를 리턴함
  - Char 배열의 경우 문자의 끝에 문자열의 끝을 알리는 NULL 문자('₩0')가 들어 있어야 함, strlen의 경우 NULL 문자를 기준으로 문자 개수를 측정함

5-help slide (2/2)

■ **B) 비교하는 부분을 직접 작성한다.**

```
#include <cstring>

RelationType StrType::ComparedTo(StrType otherString)
{
      배열 첨자를 위한 변수 idx 선언 및 0으로 초기화
      while(둘 중 하나가 문자열의 끝에 도달하기 전까지 && 두 문자가 같을 때)
      {
            ... // 두 스트링의 idx번째  원소를 서로 비교?
      }
      ... //while문에서 검출되지 않는 상황에 대하여, 어떤 문자열이 더 큰지 비교.

}
```