

객체지향프로그래밍 LAB #10

<기초문제>

1. 아래의 프로그램을 작성하시오. (*구현* 부분을 채울 것, 표의 상단: 소스코드, 하단: 실행결과)

```
#include <iostream>
#include <string>
using namespace std;

class Point {
private: // class 안에서만 사용가능
    int x; // 멤버 변수
    int y;

public: // class 안/밖에서 사용가능

    // Point() {}
    /*구현*/ { //constructor: class와 이름이 같다,
        x = _x;
        y = _y;
    }

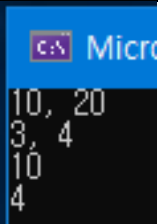
    /*X,Y set함수 구현(함수 1개)*/

    /*X,Y get 함수 각각 구현(함수 2개)*/

    void print() {
        cout << x << ", " << y << endl;
    }
};

int main() {
    Point pt1(1, 2), pt2(3, 4);
    pt1.setXY(10, 20);
    pt1.print();
    pt2.print();

    cout << pt1.getX() << endl;
    cout << pt2.getY() << endl;
    return 0;
}
```



```
10, 20
3, 4
10
4
```

2. 아래의 프로그램을 작성하시오. (/구현/ 부분을 채울 것)

```
#include <string>
#include <iostream>
using namespace std;

class Account {
//private: //멤버변수
    string name;
    string id;
    double balance;
public: // 멤버함수(method)
    // 3개의 생성자 구현
    /*구현*/ // 기본생성자. name : "", id : "", balance : 0 으로 초기화
    /*구현*/ // name, id 받아오고, balance는 0으로 초기화
    /*구현*/ // name, id , balance 받아와서 초기화, balance < 0 인 경우 0으로 초기화
    void deposit(double _amt) { balance += _amt; }
    bool withdraw(double _amt) {
        if (balance - _amt < 0)
            return false;
        balance -= _amt;
        return true;
    }
    void print() {
        cout << name << ", " << id << ", " << balance << endl;
    }
};

int main() {
    Account ac1("배성호", "1002", 5000);
    Account ac2;
    ac2.print();
    ac1.print();

    int depo;
    cout << "예금할 금액을 입력하세요 : ";
    cin >> depo;
    ac1.deposit(depo);
    /*구현*/ // print() 함수로 ac1 출력

    int wdrw;
    cout << "출금할 금액을 입력하세요 : ";
    cin >> wdrw;
    if (!ac1.withdraw(wdrw))
        cout << "잔액이 부족합니다." << endl;
    ac1.print();
    return 0;
}
```

```
, , 0
배성호, 1002, 5000
예금할 금액을 입력하세요 : 20000
배성호, 1002, 25000
출금할 금액을 입력하세요 : 500
배성호, 1002, 24500
```

```
, , 0
배성호, 1002, 5000
예금할 금액을 입력하세요 : 500
배성호, 1002, 5500
출금할 금액을 입력하세요 : 7000
잔액이 부족합니다.
배성호, 1002, 5500
```

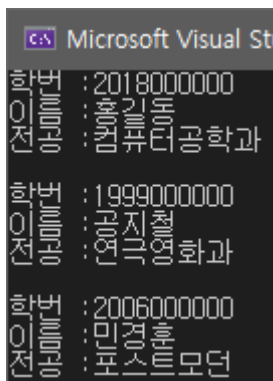
<응용문제>

1. 주어진 main 함수는 학번, 이름, 전공을 입력받아 출력한다. 주어진 코드를 기반으로 아래의 조건에 맞게 CStudent 클래스를 작성하시오.

- A. 기본 생성자는 initialization list를 이용하여 초기화함.
- B. 학번, 이름, 전공을 파라미터로 받아 초기화하는 생성자도 작성함.
- C. 학번, 이름, 전공 각각에 대해 파라미터로 받아서 설정하는 멤버함수 및 클래스 외부로 해당 멤버 변수들을 가져올 수 있도록 하는 멤버함수 설정.
- D. 멤버함수 Display는 출력화면과 같이 화면에 출력해야 함.
- E. 학번(int), 이름(string), 전공(string) 각각 멤버변수가 존재하고 private으로 설정.

```
int main() {  
    CStudent s1; // A  
    s1.Display();  
  
    CStudent s2(1999000000, "공지철", "연극영화과"); // B  
    s2.Display();  
  
    // C  
    s1.setNumber(2006000000);  
    s1.setName("민경훈");  
    s1.setMajor("포스트모던");  
    cout << "학번 : " << s1.getNumber() << endl;  
    cout << "이름 : " << s1.getName() << endl;  
    cout << "전공 : " << s1.getMajor() << endl;  
  
    return 0;  
}
```

1-출력화면:



```
Microsoft Visual Studio  
학번 : 2018000000  
이름 : 홍길동  
전공 : 컴퓨터공학과  
  
학번 : 1999000000  
이름 : 공지철  
전공 : 연극영화과  
  
학번 : 2006000000  
이름 : 민경훈  
전공 : 포스트모던
```

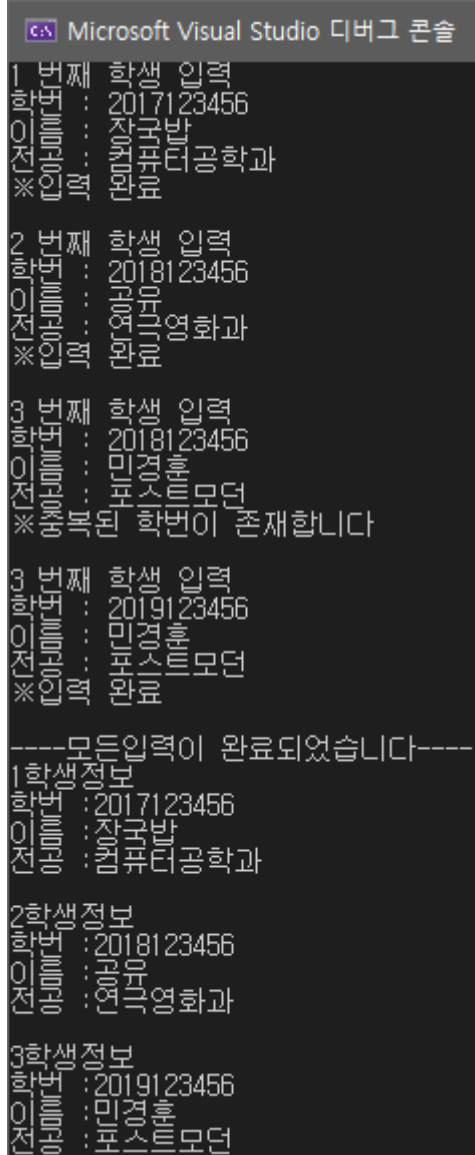
2. 1번 문제에서 작성한 CStudent 클래스를 사용하여 최대 3명의 학생 정보를 입력받고 출력하는 프로그램을 작성하시오. 단, 학번은 고유하기 때문에 Primary key로 설정하고 중복되지 않도록 구현함.

```
int main() {
    CStudent s[3]; // 3명의 학생정보를 저장할 class 배열
    int inputNumber; // 키보드로 학번을 입력 받을 변수
    string inputName, inputMajor; // 키보드로 이름, 전공을 입력 받을 변수
    int length = 0; // 현재 입력된 학생의 수

    while (1) { /* 구현 */ }

    return 0;
}
```

2-출력화면:



```
Microsoft Visual Studio 디버그 콘솔
1 번째 학생 입력
학번 : 2017123456
이름 : 장국밥
전공 : 컴퓨터공학과
※입력 완료

2 번째 학생 입력
학번 : 2018123456
이름 : 공유희
전공 : 연극영화과
※입력 완료

3 번째 학생 입력
학번 : 2018123456
이름 : 민경훈
전공 : 포스트모던
※중복된 학번이 존재합니다

3 번째 학생 입력
학번 : 2019123456
이름 : 민경훈
전공 : 포스트모던
※입력 완료

----모든입력이 완료되었습니다----
1학생 정보
학번 : 2017123456
이름 : 장국밥
전공 : 컴퓨터공학과

2학생 정보
학번 : 2018123456
이름 : 공유희
전공 : 연극영화과

3학생 정보
학번 : 2019123456
이름 : 민경훈
전공 : 포스트모던
```

3. 다음에 따라 학사관리 프로그램을 작성하시오.

1) Student 클래스 멤버변수

- 이름(string), 학번(int), 학과(string)를 멤버변수로 생성함.
- 학생이 들은 과목(vector<string>)과 성적(vector<char>)을 저장하는 멤버변수를 생성함.
- 모든 멤버변수는 private으로 설정함.

2) Student 클래스의 멤버함수들은 모두 public으로 생성하시오. 아래와 같이 두 가지 방식으로 main 함수에서 Student 객체를 생성할 수 있도록 생성자를 만드시오.

- Student Harry("Harry", 2018101234, "SWCON");
 - Student Ron;
- // 이때 Initialization list를 이용하여 초기값("default", 0, "depart")을 설정하시오.

3) Student Ron과 같은 방식으로 객체를 생성한 경우, 추후 이름, 학번, 학과를 세팅할 수 있도록 setName, setID, setDpt 함수를 각각 만드시오. Student 객체의 정보를 다음과 같이 한 줄에 출력할 수 있도록 print 함수를 만드시오.

```
Harry 2017310973 CS
```

4) 학생의 성적을 입력하는 addGrade 함수를 구현하시오. 이 함수는 과목(string) 하나와 해당 과목의 성적(char)을 파라미터로 받아 과목 이름은 과목 vector 가장 마지막에, 성적은 성적 vector 가장 마지막에 저장한다. 과목과 성적 정보를 출력하는 printGrades 함수를 구현하시오. 이 함수는 학생의 성적을 vector에 저장된 만큼 다음과 같이 한 줄에 한 과목씩 출력한다.

```
Computer Architecture B
Machine Learning B
Computer Vision C
```

5) 학생의 성적을 평균내어 평점을 구하는 getGPA 함수를 구현하시오. 이 함수는 성적 vector에 있는 A, B, C, D, F를 4, 3, 2, 1, 0으로 변환하여 모두 더해 평균을 내서 float으로 반환하는 함수이다.

- printGrades 함수 마지막 부분에 추가해서, 과목 이름과 성적 출력 후, 마지막에 평균 평점을 출력하도록 하라.

```
Computer Architecture B
Machine Learning B
Computer Vision C
GPA : 2.66667
```

6) 학번을 보고 입력받은 해를 기준 몇 학년 인지 출력하는 getYear함수를 구현하시오. 이 함수는 year(int) 정보를 받아 Freshmen(1학년), Sophomore(2학년), Junior(3학년), Senior(4학년), About to graduate(5학년 이상) 중 하나를 출력한다.

- 휴학 등은 고려하지 않고, 입력 받은 년도와 학번의 차이로만 학년을 계산한다.

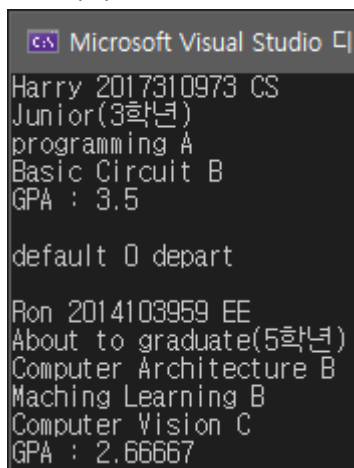
7) main 함수는 다음과 같이 작성한다.

```
int main() {
    Student Harry("Harry", 2017310973, "CS");
    Harry.print();
    Harry.getYear(2019);
    Harry.addGrade("programming", 'A');
    Harry.addGrade("Basic Circuit", 'B');
    Harry.printGrades();
    cout << "WnWn";

    Student Ron;
    Ron.print();
    cout << "Wn";
    Ron.setName("Ron");
    Ron.setID(2014103959);
    Ron.setDpt("EE");
    Ron.print();
    Ron.getYear(2019);
    Ron.addGrade("Computer Architecture", 'B');
    Ron.addGrade("Maching Learning", 'B');
    Ron.addGrade("Computer Vision", 'C');
    Ron.printGrades();
    cout << "WnWn";

    return 0;
}
```

3-출력화면:



```
Microsoft Visual Studio 디
Harry 2017310973 CS
Junior(3학년)
programming A
Basic Circuit B
GPA : 3.5

default 0 depart

Ron 2014103959 EE
About to graduate(5학년)
Computer Architecture B
Maching Learning B
Computer Vision C
GPA : 2.66667
```

(Optional)

□ 해당 부분은 성적 산출에 반영되지 않음

기반 지식

우리는 동적 메모리 할당, 레퍼런스와 포인터, 그리고 C 스타일의 문자열(char 배열)을 배웠다.

문제

문제 개요

기존에 제작했던 문자열을 다루는 함수를 보고 만족한 개발팀에서 이번엔 `std::string`처럼 동적으로 문자의 길이를 변환해주는 함수를 문의했다. 이에 앞으로 개발팀에서 모든 문자열은 동적할당해서 사용할 것이라고 약속을 잡았다. 당신에게 주어진 임무는

1. 문자열을 동적으로 생성하는 동시에 문자를 넣는 함수
2. 한 문자열의 크기를 주어진 만큼 늘려주는 함수
3. 두 문자열을 합쳐 새로운 문자열을 반환하는 함수
4. 한 문자열 안에 다른 문자열이 속해있을 경우 해당 내용과 공간을 줄여주는 함수

를 생성하는 것이다.

문제 설명

5. `char* create_cstring(/* 입력변수 직접 구현 */);`
6. `void reserve(/* 입력변수 직접 구현 */);`
7. `char* add_cstring_dyn(/* 입력변수 직접 구현 */);`
8. `void remove_substring(/* 입력변수 직접 구현 */);`
- 9.

문제 규칙

`iostream`의 `cout cin`을 제외한 외부 라이브러리(STL 포함)의 사용을 불허한다.

함수 내에서 메모리 누수가 나서는 안된다.

예제 코드

```
// my_string_dyn.h
#pragma once
```

```

char* create_cstring( /* 입력변수 직접 구현 */ );
void reserve( /* 입력변수 직접 구현 */ );
char* add_cstring_dyn( /* 입력변수 직접 구현 */ );
void remove_substring( /* 입력변수 직접 구현 */ );

// main.cpp
#include "my_string_dyn.h"

int main()
{
    char* hello = create_cstring("Hello");    // hello는 "Hello"를 내용물
로 가짐
    char world_arr[6] = "World";
    char* world = create_cstring(world_arr);  // world는 "World"를 내용물
로 가짐

    char* hello_world = new char[5];
    reserve(hello_world, 255);    // hello_world는 총 255 바이트의 용량을
가짐
    reserve(hello, 10);    // hello는 "Hello"라는 내용물과 10바이트의 용량
을 가짐

    char* new_cstring = new char[10];
    new_cstring = add_cstring_dyn(hello, world);    // new_cstring의 내용
물은 "HelloWorld", 용량은 11바이트
    new_cstring = add_cstring_dyn(new_cstring, "!");    // new_cstring의
내용물은 "HelloWorld!", 용량은 12바이트

    remove_substring(new_cstring, hello);    // new_cstring의 내용물은
"World!", 용량은 7바이트
    remove_substring(new_cstring, "orld");    // new_cstring의 내용물은
"W!", 용량은 3바이트
    remove_substring(new_cstring, new_cstring);    // new_cstring의 내용
물은 "", 용량은 1바이트

    return 0;
}

```