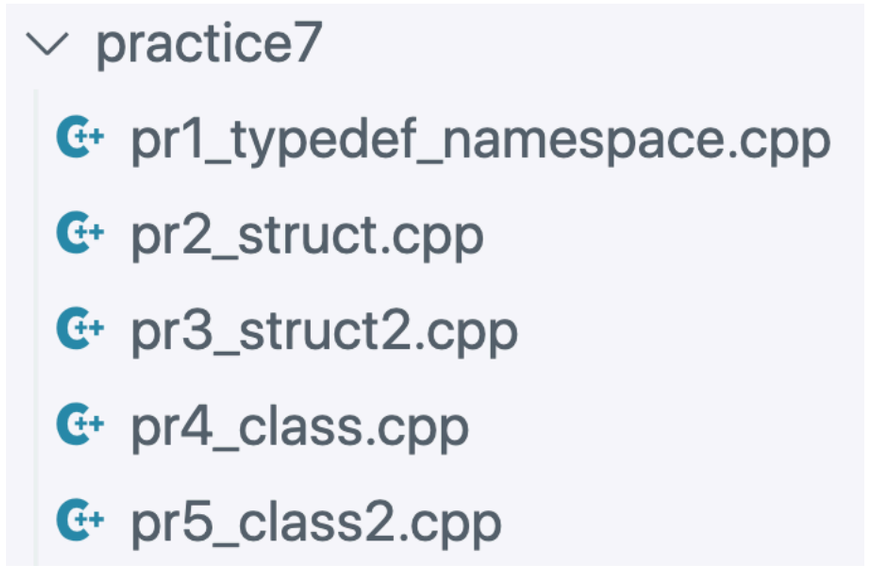
C++ 실습 6

실습 내용

- 이번주는 총 5개의 구조체와 클래스에 관련된 실습을 진행합니다.

- Git Pull로 새로운 파일들을 받으세요.



첫번째 실습

- 첫번째 실습은 typedef 명령과 namespace에 관련된 내용입니다.

- 언제나와 다르게 시작 부분에 using namespace std가 없어진 것을 확인하세요.

- 대신 두 개의 중첩된 namespace가 정의되어 있습니다.

```cpp
#include<iostream>
#include<cstdint>

namespace my_func {
    int atoi(const char* s) {
        return std::atoi(s) + 1;
    }
    namespace extra {
        int atoi(const char* s) {
            return std::atoi(s) + 10;
        }
    }
}
```

첫번째 실습

```cpp
typedef int size_t;
typedef double ratio_t;
using score_t = double;

size_t arr_size = 100;
ratio_t screen_ratio = 0.3;
score_t my_score = 4.0;

typedef int int_arr_t[2]; //[] 위치에 주의.
int_arr_t arr[3] = {{1, 2}, {2, 3}, {3, 4}}; //3x2 배열.
```

```cpp
int16_t int1 = 11; //int16_t를 ctrl+click하여 정의를 확인.
int32_t int2 = 111;
std::cout << "int16_t size: " << sizeof(int16_t) << std::endl;
std::cout << "int32_t size: " << sizeof(int32_t) << std::endl;
```

- 전반부는 typedef에 관련된 실습입니다.
- 강의시간에 보았던 예제가 있으니 확인해 보세요.
- 마지막 부분은 플랫폼 독립적인 정수형과 관련된 예제입니다.
- int16_t를 ctrl + click하면 새로 파일이 열리고 typedef로 정의된 것을 확인할 수 있습니다.

첫번째 실습

```
//Namespace
int num = std::atoi("999") + 1;
int num2 = my_func::atoi("999") + 1;
std::cout << "num = " << num << std::endl;
std::cout << "num2 = " << num2 << std::endl;

int num3 = my_func::extra::atoi("999") + 1;
namespace ext = my_func::extra;
int num4 = ext::atoi("999") + 1;
std::cout << "num3 = " << num3 << std::endl;
std::cout << "num4 = " << num4 << std::endl;
```

- 후반부는 namespace에 관련된 내용으로 서로 다른 namespace에 정의된 atoi()함수를 호출하여 결과를 비교하는 부분입니다.

- cout과 endl도 평소와 다르게 std::cout 등으로 지정해서 호출해야 하는 것을 확인할 수 있습니다.

두번째 실습

- 두번째 실습은 구조체 관련 실습입니다.
- 첫번째 부분은 구조체 정의와 참조에 관한 예제입니다.
- 단순하게 구조체를 선언하고, '.'연산자를 이용 필드에 접근하는 것을 확인해 볼 수 있습니다.

```
struct Person {
    int id; //field1
    string name; //field2
    int age; //field3
}; //;를 잊지 말 것.
```

```
//Field에 값 지정.
Person dva;
dva.id = 1000;
dva.name = "Song Hana";
dva.id += 100;

cout << "D.VA's ID: " << dva.id << endl;
cout << "D.VA's Name: " << dva.name << endl;
cout << "D.VA's Age: " << dva.age << endl;
```

두번째 실습

- 두번째 실습은 후반부는 필드를 초기화하는 다른 방법을 확인할 수 있습니다.

- PetShop 구조체 부분에서는 중첩된 구조체를 어떻게 접근할 수 있는지 보여줍니다.

```
Person genji = {1111, "Genji", 35};
//C++11
Person winston {9999, "Winston"}; //값이 없으면 0으로 초기화.

cout << "Age of " << winston.name;
cout << " is " << winston.age << endl;

struct PetShop {
    Person owner;
    string address;
};

PetShop store = { winston, "Gibraltar"};
cout << "Store Owner's name is ";
cout << store.owner.name << endl;
```

세번째 실습

- 후반부에서는 구조체 포인터와 함수에서 구조체를 사용하는 예제를 확인할 수 있습니다.
- 일반적인 포인터와 동일하게 동작하고, 참조시 ->를 사용하는 것을 명심하세요.
- 함수로 호출할 때도 구조체를 이용하면 동시에 name, age를 넘겨받는 것이 가능합니다.

```
Person clone(Person p) {
    return { p.id+1000, p.name, 1 };
}

//구조체 포인터
Person keanu = { 1, "Keanu Reeves", 56 };
Person* neo = &keanu;
cout << "His name is " << keanu.name << endl;
cout << "Neo's real name is " << neo->name << endl;

//함수로 호출
Person newPerson = clone(keanu);
cout << "New person is also " << newPerson.name << endl;
cout << "But his age is " << newPerson.age << endl;
```

네번째 실습

- 네번째 실습은 클래스와 관련된 첫 번째 예제입니다.

- 강의에서 보았던 기본 생성자와 매개변수를 사용한 생성자를 사용하는 것을 확인할 수 있습니다.

- 다음장에 나머지 부분이 이어집니다.

```
class Person {
    private:
        int id;
        string na
        int age;

public:
    Person()
        id = 0;
        name = "";
        age = 0;
    }

    Person(string name, int age) {
        id = 0;
        this->name = name;
        this->age = age;
    }
```

```
//잘못된 생성자.
Person(int id, string name, i
    id = id;
    name = name;
    age = age;
}

void printName() {
        cout << "My name is " <<
    }
}; //;를 잊지 말 것.
```

네번째 실습

- 지난장에서 이어지는 클래스의 정의부분 입니다.
- 마지막 부분에는 기본 생성자 및 매개변수가 있는 생성자를 이용 값을 대입하는 예제가 있습니다.
- 또 printName() 멤버함수를 호출하는 부분을 확인할 수 있습니다.
- 아무런 인자를 넘기지 않았음에도 두 함수의 호출이 달라지는 것을 확인해 보세요.

```
//잘못된 생성자.
Person(int id, string name, int age) {
    id = id;
    name = name;
    age = age;
}

    void printName() {
        cout << "My name is " << name << endl;
    }
}; //;를 잊지 말 것.


  Person p;   //기본 생성자.
  p.printName();

  Person mcCree("Jesse McCree", 37); //매개변수 이용.
  mcCree.printName();
```

다섯번째 실습

- 다섯번째 실습은 클래스 관련 예제 두번째입니다.
- 첫번째 생성자는 id 멤버 초기화를 하는 것을 보여줍니다.
- 두번째 생성자는 생성자 위임으로 첫번째 생성자를 부르는 것을 보여줍니다.
- 마지막 부분에는 소멸자가 추가되어 있습니다.
- 각각에서는 화면에 메시지를 출력합니다.

```
class Person {
    private:
        const int id;
        string name;
        int age = 19;

    public:
        Person(int id, string name) : id(id) {
            this->name = name;
            cout << "1st Constructor!" << endl;
        }

        Person(int id, string name, int age) : Person(id, name) {
            this->age = age;
            cout << "2nd Constructor!" << endl;
        }

        void print() {
            cout << "I'm " << name << ", and I'm ";
            cout  << age << " years old." << endl;
        }

        ~Person() {
            cout << "Destructor!! - " << name << endl;
        }
}; //;를 잊지 말 것.
```

다섯번째 실습

- 앞 장의 클래스 정의를 이용하여 객체를 선언하는 부분입니다.

- dva, mcCree 객체를 선언하고 생성자가 실행될 때마다 출력되는 메시지가 어떤 순서로 나오는지 확인해 보세요.

- delete p를 실행하면 소멸자가 바로 실행되는 것을 알 수 있습니다.

- 함수 실행이 종료되는 시점에는 mcCree, dva의 순으로 소멸자가 실행됩니다.

- **이는 정적할당된 변수는 선언된 순서의 역순으로 객체가 소멸하기 때문입니다.**

```
//기본 생성자가 없으므로 반드시 매개변수 필요.
Person dva(111, "Song Hana");
dva.print();

Person mcCree(123, "Jesse McCree", 37);
mcCree.print();

//포인터로 동적할당.
Person* p = new Person(101, "Dying Man", 99);
p->print();
delete p;
```

다섯번째 실습

- 다섯번째 실습 후반부에서는 새로운 클래스 Person2를 사용합니다.

- Person2 클래스는 생성자에서 매개변수의 기본값을 지정하였습니다.

- 실제 객체를 선언할 때 이 기본값이 어떻게 동작하는지 확인해 보세요.

```
class Person2 {
    private:
        int id;
        string name;
        int age;

    public:
        Person2(int id = 0, string name = "", int age = 1) {
            this->id = id;
            this->name = name;
            this->age = age;
        }
        void print() {
            cout << "ID: " << id;
            cout << "\tName: " << name;
            cout << "\tAge: " << age << endl;
        }
};
```

```
Person2 p1 (111);
Person2 p2 (123, "Another Person");
p1.print();
p2.print();
```

실습 제출물

- 다섯번째 실습의 슬라이드 12에 있는 코드가 실행되면서 생성자와 소멸자에서 메시지가 출력되는 부분을 캡쳐하여 제출하시면 됩니다.

- 이번에는 메시지가 좀 많으므로, 오른쪽과 같이 출력된 메시지 부분만 캡쳐하여 제출하세요.

- 전체가 다 보이지 않는 경우 마우스로 터미널 부분의 크기를 조절해야 할 수 있습니다.

```
1st Constructor!
I'm Song Hana, and I'm 19 years old.
1st Constructor!
2nd Constructor!
I'm Jesse McCree, and I'm 37 years old.
1st Constructor!
2nd Constructor!
I'm Dying Man, and I'm 99 years old.
Destructor!! - Dying Man
ID: 111 Name:   Age: 1
ID: 123 Name: Another Person    Age: 1
Destructor!! - Jesse McCree
Destructor!! - Song Hana
```

실습 정리

- 이번주는 총 5개의 파일로 실습을 진행하였습니다.
- typedef, namespace의 기본적인 사용법을 확인했습니다.
- 구조체를 정의하고 필드에 접근하는 법을 실습하였습니다.
- 클래스와 객체의 생명주기와 관련된 실습을 진행하였습니다.
- 특히 객체의 생명주기 부분은 주의깊게 확인하시기 바랍니다.