



C++ - 모듈 07

C++ 템플릿

요약:

이 문서에는 C++ 모듈 중 모듈 07의 연습 문제가 포함되어 있습니다.

버전: 6

콘텐츠

I	소개	2
II	일반 규칙	3
III	연습 00: 몇 가지 기능으로 시작하기	5
IV	연습 01: 반복	7
V	연습 02: 배열	8

1장 소개

C++는 비야른 스트로스트룹이 C 프로그래밍 언어 또는 '클래스가 있는 C'의 변형으로 만든 범용 프로그래밍 언어입니다(출처: [위키백과](#)).

이 모듈의 목표는 **객체 지향 프로그래밍**을 소개하는 것입니다. 이것이 C++ 여정의 시작점이 될 것입니다. OOP를 배우기 위해 많은 언어가 권장됩니다. C++는 여러분의 오랜 친구인 C에서 파생된 언어이기 때문에 C++를 선택하기로 결정했습니다. C++는 복잡한 언어이며, 코드를 단순하게 유지하기 위해 C++98 표준을 준수합니다.

최신 C++는 여러 측면에서 많이 다르다는 것을 알고 있습니다. 따라서 능숙한 C++ 개발자가 되고 싶다면 42개의 공통 코어를 넘어서는 것은 여러분에게 달려 있습니다!

2장 일반 규칙

컴파일

- c++와 -Wall -Wextra -Werror 플래그를 사용하여 코드를 컴파일합니다.
- 플래그 -std=c++98을 추가하면 코드가 계속 컴파일됩니다.

서식 및 이름 지정 규칙

- 연습 디렉터리의 이름은 ex00, ex01, ..., exn
- 가이드라인에서 요구하는 대로 파일, 클래스, 함수, 멤버 함수 및 속성의 이름을 지정하세요.
- 클래스 이름을 **대문자 대소문자** 형식으로 작성합니다. 클래스 코드가 포함된 파일은 항상 클래스 이름에 따라 이름이 지정됩니다. 예를 들어 ClassName.hpp/ClassName.h, ClassName.cpp 또는 ClassName.hpp. 예를 들어, 벽돌 벽을 의미하는 "BrickWall" 클래스의 정의가 포함된 헤더 파일이 있다면 그 이름은 BrickWall.hpp가 됩니다.
- 달리 지정하지 않는 한, 모든 출력 메시지는 새 줄 문자로 끝나야 하며 표준 출력에 표시되어야 합니다.
- *안녕, 노미네트!* C++ 모듈에는 코딩 스타일이 강제되지 않습니다. 좋아하는 스타일을 따를 수 있습니다. 하지만 동료 평가자가 이해할 수 없는 코드는 채점할 수 없는 코드라는 점을 명심하세요. 깔끔하고 읽기 쉬운 코드를 작성하기 위해 최선을 다하세요.

허용/금지

더 이상 C로 코딩하지 마세요. 이제 C++로 전환하세요! 그러므로:

- 표준 라이브러리의 거의 모든 것을 사용할 수 있습니다. 따라서 이미 알고 있는 것을 고수하는 대신 익숙한 C 함수의 C++ 버전을 최대한 많이 사용하는 것이 현명할 것입니다.
- 하지만 다른 외부 라이브러리는 사용할 수 없습니다. 즉, C++11(및 파생 형식) 및 Boost 라이브러리는 금지됩니다. 다음 함수도 금지됩니다: `*printf()`, `*alloc()` 및 `free()`. 이 함수를 사용하면 성적은 0점이 됩니다.

- 명시적으로 달리 명시되지 않는 한 네임스페이스 <ns_name> 및 친구 키워드는 금지되어 있습니다. 그렇지 않으면 성적은 -42점이 됩니다.
- **모듈 08과 09에서만 STL을 사용할 수 있습니다.** 즉, 그때까지는 컨테이너(벡터/리스트/맵 등)와 알고리즘(<알고리즘> 헤더를 포함해야 하는 모든 것)을 사용할 수 없습니다. 그렇지 않으면 성적이 -42점이 됩니다.

몇 가지 설계 요구 사항

- 메모리 누수는 C++에서도 발생합니다. 메모리를 할당할 때(새로운 키워드)의 경우 **메모리 누수**를 방지해야 합니다.
- 모듈 02부터 모듈 09까지, **명시적으로 달리 명시된 경우를 제외하고** 클래스는 **정통 정석 양식**으로 디자인해야 합니다.
- 헤더 파일에 있는 모든 함수 구현(함수 템플릿 제외)은 연습에 0을 의미합니다.
- 각 헤더를 다른 헤더와 독립적으로 사용할 수 있어야 합니다. 따라서 필요한 모든 종속성을 포함해야 합니다. 그러나 **include 가드**를 추가하여 이중 포함 문제를 피해야 합니다. 그렇지 않으면 성적이 0점이 됩니다.

읽기

- 필요한 경우 추가 파일을 추가할 수 있습니다(예: 코드를 분할하는 경우). 이러한 과정은 프로그램에서 확인하지 않으므로 필수 파일을 제출하는 한 자유롭게 추가할 수 있습니다.
- 때때로 연습 지침이 짧아 보이지만 지침에 명시되지 않은 요구 사항이 예제에 나와 있는 경우가 있습니다.
- 시작하기 전에 각 모듈을 완전히 읽으세요! 정말 그렇게 하세요.
- 오딘의, 토르의! 머리를 써라!!!




많은 클래스를 구현해야 합니다. 자주 사용하는 텍스트 편집기를 스크립팅할 수 없다면 이 작업이 지루해 보일 수 있습니다.



운동을 완료하는 데는 어느 정도의 자유가 주어집니다. 그러나 필수 규칙을 따르고 게으르지 마세요. 유용한 정보를 많이 놓칠 수 있습니다! 이론적 개념에 대해 주저하지 말고 읽어보세요.

제3장

연습 00: 몇 가지 기능으로 시작하기

	운동 : 00
	몇 가지 기능으로 시작하세요.
	제출 디렉토리 : <i>ex00/</i>
	제출할 파일 : 메이크파일, <i>main.cpp</i> , <i>whatever.{h, hpp}</i>
	금지된 기능 : 없음

다음 함수 템플릿을 구현합니다:

- 스왑: 주어진 두 인자의 값을 바꿉니다. 아무것도 반환하지 않습니다.
- 최소: 인자로 전달된 두 값을 비교하여 가장 작은 값을 반환합니다. 두 값이 같으면 두 번째 값을 반환합니다.
- 최대: 인자로 전달된 두 값을 비교하여 가장 큰 값을 반환합니다. 두 값이 같으면 두 번째 값을 반환합니다.

이 함수는 모든 유형의 인수를 사용하여 호출할 수 있습니다. 유일한 요구 사항은 두 인수의 유형이 같아야 하고 모든 비교 연산자를 지원해야 한다는 것입니다.



템플릿은 헤더 파일에 정의해야 합니다.

다음 코드를 실행합니다:

```
int    main( void ) {

    int a = 2;
    int b = 3;

    ::SWAP( A, B );
    std::cout << "a = " << a << ", b = " << b << std::endl;
    std::cout << "min( a, b ) = " << ::min( a, b ) << std::endl;
    std::cout << "max( a, b ) = " << ::max( a, b ) << std::endl;

    STD::문자열 C = "체인1"; STD::문자열 D
    = "체인2";


    ::SWAP(C, D);
    std::cout << "c = " << c << ", d = " << d << std::endl;
    std::cout << "min( c, d ) = " << ::min( c, d ) << std::endl;
    std::cout << "max( c, d ) = " << ::max( c, d ) << std::endl;

    반환값은 0입니다;
}
```

출력해야 합니다:

```
a = 3, b = 2
min(a, b) = 2
max(a, b) = 3
c = 체인2, d = 체인1
min(c, d) = 체인1 max(c,
d) = 체인2
```


4장 연습 01: 반복

	운동 : 01
Iter	
체크인 디렉토리 : <code>ex01/</code>	
제출할 파일 : 메이크파일, <code>main.cpp</code> , <code>iter.{h, hpp}</code>	
금지된 기능 : 없음	


3개의 매개변수를 받고 아무것도 반환하지 않는 함수 템플릿 반복을 구현합니다.

- 첫 번째 매개변수는 배열의 주소입니다.
- 두 번째는 배열의 길이입니다.
- 세 번째는 배열의 모든 요소에서 호출되는 함수입니다.

테스트가 포함된 `main.cpp` 파일을 제출합니다. 테스트 실행 파일을 생성할 수 있는 충분한 코드를 제공합니다.

반복 함수 템플릿은 모든 유형의 배열에서 작동해야 합니다. 세 번째 매개변수는 인스턴스화된 함수 템플릿일 수 있습니다.

5장 연습 02: 배열

	운동 : 02
배열	
제출 디렉토리 : <i>ex02/</i>	
제출할 파일 : <i>Makefile, main.cpp, Array.{h, hpp}</i> 및 선택적 파일 : <i>Array.tpp</i>	
금지된 기능 : 없음	

유형 *T*의 요소를 포함하고 다음 동작과 함수를 구현하는 클래스 템플릿 **Array**를 개발합니다:

- 매개변수가 없는 구조체: 빈 배열을 생성합니다.
- 부호 없는 정수 *n*을 매개변수로 사용하는 구조체입니다: 기본적으로 초기화된 *n*개의 엘리먼트로 구성된 배열을 생성합니다.
팁: `int * a = new int();` 를 컴파일한 다음 `*a`를 표시하세요.
- 복사 및 할당 연산자에 의한 구성. 두 경우 모두 복사 후 원본 배열이나 복사본을 수정해도 다른 배열에 영향을 미치지 않습니다.
- 메모리를 할당할 때는 반드시 `new[]` 연산자를 사용해야 합니다. 예방적 할당(메모리를 미리 찾아내는 것)은 금지되어 있습니다. 프로그램은 할당되지 않은 메모리에 절대로 접근해서는 안 됩니다.
- 요소는 아래 첨자 연산자를 통해 액세스할 수 있습니다: `[]`.
- 연산자를 사용하여 요소에 액세스할 때 해당 인덱스가 범위를 벗어난 경우 `std::예외`가 발생합니다.
- 배열의 요소 수를 반환하는 `size()` 멤버 함수입니다. 이 멤버 함수는 매개변수를

받지 않으며 현재 인스턴스를 수정하지 않습니다.

평소와 같이 모든 것이 예상대로 작동하는지 확인하고 테스트가 포함된 `main.cpp` 파일을 제출하세요.