과정명 : Modern C++ (C++11/14/17)

문제	보기	NULL POINTER 를 나타내는 리터럴은 무엇인가요 ?
1	1	null
	2	NULL
	3	nullptr
	4	zero

문제	보기	다음중 에러는 ? (int n = 10 일때)
2	1	const int c1 = 10;
	2	const int c2 = n;
	3	constexpr int c1 = 10
	4	constexpr int c2 = n;

문제	보기	복사 생성자가 아닌 이동 생성자를 호출하기 위해 사용하는 C++ 표준 함수는 ?
3	1	std::move
	2	std::forward
	3	std::copy
	4	std::construct_at

문제	보기	아래 코드에서 auto 는 각각 무슨 타입으로 결정 될까요 ?
		int x[3] = {1,2,3};
		auto a1 = x; // auto 의 타입은 ?
		auto& a2 = x; // auto 의 타입은 ?

4	1	int[3], int[3]
	2	int*, int[3]
	3	int*, int*
	4	int[3], int*

문제	보기	객체 초기화시에 직접 초기화만 가능하고 복사 초기화는 사용할 수 없도록 하기위해 생성자 앞에 붙이는 키워드는 무엇인가요 ?
5	1	final
	2	constexpr
	3	default
	4	explicit

문제	보기	C++11 에서 추가된 키워드로 가상함수 재정의시 오타등의 실수를 방지하기 위해 사용하는 키워드는 ?
6	1	final
	2	override
	3	const
	4	default

문제	보기	다음 설명중 틀린 것은 ?
7	1	generic lambda expression 을 사용하면 operator() 연산자를 템플릿으로 만들수 있다.
	2	람다 표현식은 함수 객체를 만드는 표현식이다.
	3	mutable 람다는 값으로 캡쳐된 지역변수의 원본을 수정할 수 있다.
	4	복사 생성자를 사용할 수 없게 하려면 함수를 삭제 하면 된다.

문제	보기	Forwarding reference 로 전달 받은 인자를 다른 곳으로 전달할 때 완벽한 전달을 위해 사용하는 C++ 표준 함수의 이름은 무엇인가요 ?
8	1	std::decay
	2	std::make_shared
	3	std::send
	4	std::forward

문제	보기	variadic template 의 parameter pack 안에 있는 모든 요소에 이항 연산을 수행하는 문법을 무엇이라고 하나요 ?
9	1	lambda expression
	2	fold expression
	3	unevaluated expression
	4	perfect forwarding

문제	보기	아래 코드를 보고, 주석 A, B, C, D 에 들어갈 타입으로 맞는 것을 골라 보세요
		
		template<typename T> void foo(T&& a) {}
		
		int main()
		{
		int n = 10;
		foo(n); // T의 타입 : A a의 타입 : B
		foo(10); // T의 타입 : C a의 타입 : D
		}

10	1	A : int&, B : int&, C : int, D : int&&
	2	A : int, B : int&, C : int, D : int&&
	3	A : int, B : int, C : int, D : int&&
	4	A : int&, B : int&, C : int&&, D : int&&

11	문제	함수 템플릿 사용시 치환에 실패하면 에러가 발생하지 않고 함수의 후보군에서 제외된다는 것을 나타내는 C++ Idioms 는?
	1	IFTHENELSE
	2	CRTP
	3	SFINAE
	4	TYPE TRAITS

12	문제	T가 포인터 인지 조사하려고 합니다.맞는 코드는 ?
	1	std::is_pointer<T>
	2	std::is_pointer<T>::value
	3	std::is_pointer<T>::type
	4	std::remove_pointer<T>::value

13	문제	가변인자 템플릿에서 parameter pack안에 있는 요소의 갯수를 구하는 연산자는 무엇인가?
	1	sizeof
	2	decltype
	3	declval
	4	sizeof...

14	지문	`void hoo(int a, int b, int c) {}`
		
		`int  negate(int n) { return -n; }`
		
		`template<typename ... Types> void foo(Types ... args)`
		`{`
		`  (A);`
		`}`

	문제	위 코드의(A) 부분에 놓일수 있는 코드는?
	1	hoo(negate(args));
	2	hoo(negate(args...));
	3	hoo(negate(args)...);
	4	hoo(negate(args...)...);

15	문제	다음 중 C++11 문법에 대한 설명중 잘못된 것은?
	1	typedef 는 타입의 대한 별칭을 만들지만 using을 사용하면 타입뿐 아니라 template에 대한 별칭도 만들수 있다.
	2	assert()는 실행시간 유효성 확인만 할수 있지만 static_assert를 사용하면 실행시간과 컴파일 시간 유효성을 모두 확인할 수 있다.
	3	함수가 예외가 없음을 나타날 때는 noexcept를 사용한다.
	4	nullptr 은 포인터 0을 의미 한다.

16	문제	다음 중 에러가 발생하는 것은?
	1	char c1{ 0 };
	2	char c2 = { 0 };
	3	char c3 = { 'a' };
	4	char c4 = { 300 };

17	지문	`template<typename T> struct ResultType`
		`{`
		`};`
		
		`template<typename T1, typename T2> struct ResultType< A >`
		`{`
		`  typedef T1 type;`
		`};`

	문제	인자가 한 개인 함수의 리턴 타입을 구하는 traits 를 만들려고 한다. (A)에 들어갈 표현중 맞는 것은 ?
	1	T1, T2
	2	T1(T2)
	3	void(T1, T2)
	4	T2, T1

18	지문	class Test
		{
		public:
		Test(int a, int b) { cout << "1" << endl; }
		Test(initializer_list<int>) { cout << "2" << endl; }
		};
		int main()
		{
		Test t1(1, 1);
		Test t2{ 1, 1 };
		Test t3 = { 1,1 };
		}

	문제	위 코드의 수행결과로 맞는 것을 고르시오.
	1	1 2 2
	2	1 1 1
	3	1 1 2
	4	2 2 2

19	지문	int main()
		{
		int x[3] = { 1,2,3 };
		
		auto a1 = x[0]; // 1
		decltype(x[0]) d1; / 2
		auto a2 = x; // 3
		decltype(auto) d2 = x; // 4
		}

	문제	위 코드중 에러가 발생하는 것은?
	1	1, 2
	2	2, 3
	3	2, 4
	4	3, 4

20	지문	template<typename T> void foo(T&& arg)
		{
		decltype(std::move(arg)) a;
		decltype(std::forward<T>(arg)) b;
		}
		int main()
		{
		int n = 10;
		
		foo(n);
		foo(10);
		}

	문제	위 코드가 실행 되었을 때 a, b의 타입으로 맞는 것은?

1	foo(n) 일때 a : int&&, b : int&
2	foo(n) 일때 a : int&, b : int&
3	foo(10) 일때 a : int&&, b : int&
4	foo(10) 일때 a : int&, b:int&&