프로그래밍 언어론 보고서 #4

20154068

우동범

‘

**6장 복습문제**

**1. 서술자란 무엇인가?**

서술자(descriptor)는 변수의 속성들의 모임이다. 구현에서, 서술자는 변수 속성들을 저장하는 메모리 셀들의 모임이다.속성들이 모두 정적이면, 서술자는 단지 컴파일 시간에만 필요하다.서술자는 컴파일러에 의해서 보통 심볼테이블의 일부로 구성되며, 컴파일과정에서 사용된다. 그러나 동적 속성들을 위해서 서술자의 일부나 전부가 실행 시간 동안에 유지되어야 한다, 이 경우에 서술자는 실행시간 시스템에서 사용된다. 모든 경우에 서술자는 타입 검사를 위해서 사용되고 할당 연산과 회수 연산에서 사용된다.

**9. 정적, 고정 스택-동적, 스택-동적, 고정 힢-동적, 힢-동적 배열을 정의 하시오. 각각의 장점은 무엇인가?**

**-** 정적배열 : 첨자범위가 정적으로 바인딩되고, 기억장소 할당이 정적으로 이루어지는 배열

장점 : 효율성, 즉, 동적 기억장소의 할당 및 회수가 필요하지 않다

- 고정스택-동적배열 : 첨자범위가 정적으로 바인딩되지만, 할당이 실행시간중 선언문 세련화시간에 이루어지는 배열

장점 : 정적배열에 비해서 기억장소 공간의 효율성이 좋다. 한 프로시저에 속한 대형배열이 다른 프로시저에 속한 대형 배열과 동일한 기억장소 공간을 사용할 수 있다

- 스택-동적배열 : 첨자범위가 동적으로 바인딩되고 기억장소 할당이 동적으로 이루어지는 배열이다. 그러나 일단 첨자 범위가 바인딩되고 기억장소가 할당되면, 이러한 바인딩과 할당은 그 변수의 존속기간 동안에 고정된다.

장점 : 정적배열과 고정스택-동적배열에 비해서 유연성이 좋다. 배열크기는 배열이 사용되기 전까지 알려질 필요가 없다

- 고정 힙 -동적배열 : 첨자범위가 동적으로 바인딩되고, 기억장소바인딩이 동적이나,이러한 바인딩 모두 기억공간이 할당된 후에는 고정된다는 점 에서 고정스택-동적배열과 유사하다. 차이점은 바인딩이 선언 문 세련화 시점이 아니고 사용자 프로그램이 요청하는 시점에 서 이루어진다는것과 기억공간이 힙으로부터 할당된다.

장점 : 배열의 크기가 항상 문제에 맞춤화 된다.

- 힢-동적배열 : 첨자 범위의 바인딩과 기억장소 할당이 동적으로 이루어지고, 이것 이 그 배열의 존속기간 동안에 여러번 변경될 수 있는 배열이다

장점 : 다른 유형에 비해 유연성이 좋다

**19. java의 배열 서술자에서 필요한 항목들은 무엇인가? 그리고 이들이 언제 저장되어야 하는가?(컴파일 시간 또는 실행시간)**

필요한 항목 : 배열의 이름, 크기

컴파일 시간에 저장됨.

**36. 포인터에 대한 두 가지 공통된 문제는 무엇인가?**

1. 허상 포인터 또는 허상 참조.

2. 분실된 힙 동적변수는 사용자 프로그램에서 더 이상 접근될 수 없는 할당된 힙- 동적 변수다.

**37. 대부분 언어의 포인터가 왜 한 개의 타입 변수를 가리키도록 제한되는가?**

새로운 변수가 이전 변수와 동일한 타입이 아니면 그 허상 포인터의 사용에 대한 타입검사가 유효하지 않기 때문에.

**38 C++의 참조 타입은 무엇이고, 공통된 용도는 무엇인가?**

묵시적으로 역참조되는 상수 포인터. 호출함수와 피호출 함수 간에 양방향 통신을 제공.

**39 C++의 참조 변수가 형식 매개변수에 대해서 포인터보다 더 나은 이유는 무엇인가?**

형식 매개변수는 명시적인 역 참조를 요구하지만 참조 매개변수는 다른 매개변수와 같이 피호출 함수에서 참조 되기 때문에 더 낫다.

**40 다른 언어의 포인터에 비해서 JAVA와 C#의 참조 타입 변수가 갖는 장점은 무엇인가?**

Java 참조 변수는 다름 클래스 사례들을 참조하도록 할당 될 수 있다. 즉 , 그들은 상수가 아니다.

C#은 java의 참조와 c++의 포인터 모두를 포함한다.

장점 : 참조변수가 가리키는 객체는 묵시적으로 회수되지만, 포인터가 가리키는 객체는 그렇지 않다

**41 쓰레기를 회수하는 것에 대한 지연 방법과 조기 방법을 기술하라.**

지연방법 : 실행 시간 시스템은 요구될 때 기억공간 셀들을 할당하고 필요할 때 포인터들을 셀드셀부터 분리

조기방법 : 모든 셀에 대해서 현재 자신을 가리키고 있는 포인터들의 개수를 저장하는 계수기를 유지시킴으로써 그 목적을 달성한다.

**42 JAVA와 C#의 참조에 대한 산술 연산이 의미가 없는 이유는 무엇인가?**

참조 변수는 메모리의 객체나 값을 참조해서 산술연산에 대해 의미가 없다.

**46 JAVA가 강타입이 아닌 이유는 무엇인가?**

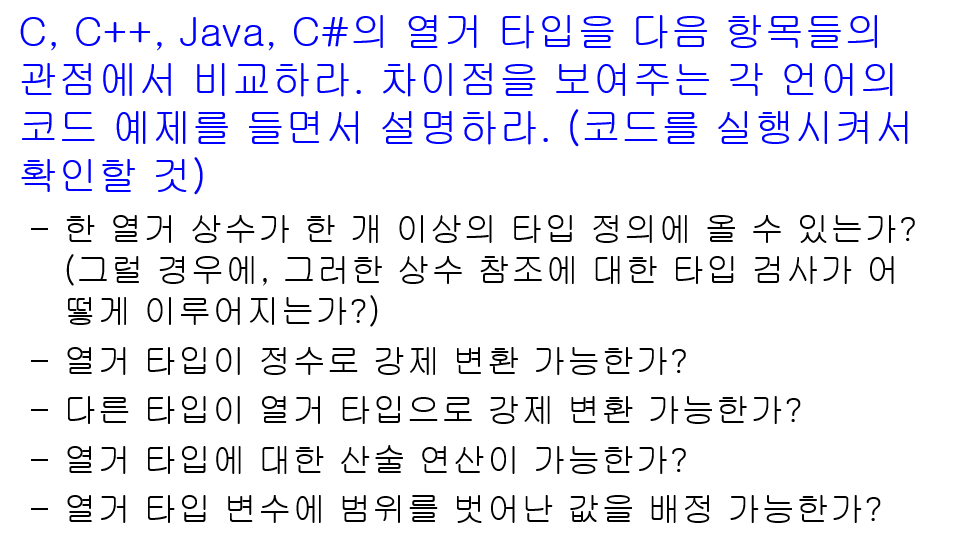
축소 변환은 허용하지 않지만 확대변환은 허용하기 때문이다.

**49 C와 C++이 강 타입이 아닌 이유는 무엇인가?**

타입검사가 되지 않는 공용체 타입을 포함하기 떄문이다. 그리고 강제변환을 많이 허용하기 때문이다.

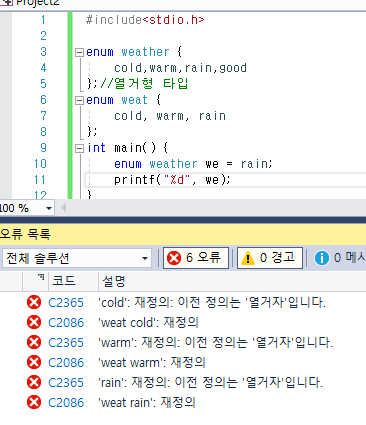
**54 C가 어떤 타입에 대해서 구조 타입 동등을 사용하는가?**

두 개의 구조체, 열거, 공용체가 다른 파일에 정의되어 있다면 이 경우에 구조 타입 동등이 사용됨.



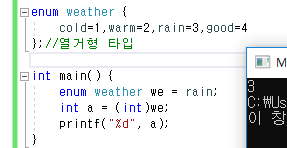
**C 언어**

- 한 열거 상수가 한 개 이상의 타입 정의에 올 수 있는가?



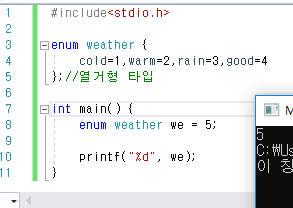
불가능 하다.

- 열거 타입이 정수로 강제 변환 가능한가?

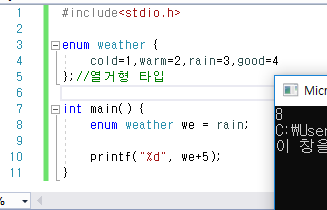


강제 변환이 가능하다.

- 다른 타입이 열거 타입으로 강제 변환 가능한가?

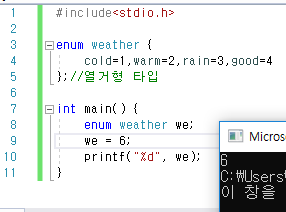
가능하다.

- 열거 타입에 대한 산술 연산이 가능한가?



Rain은 3인데 5를 더해줬더니 8을 출력하는 것을 보면 가능하다

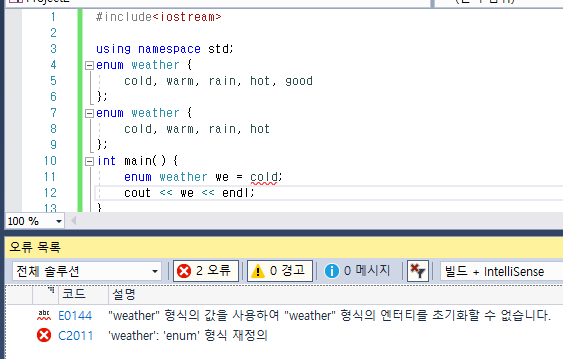
- 열거 타입 변수에 범위를 벗어난 값을 배정 가능한가?



가능 하다.

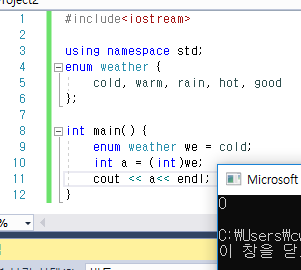
**C++**

- 한 열거 상수가 한 개 이상의 타입 정의에 올 수 있는가?



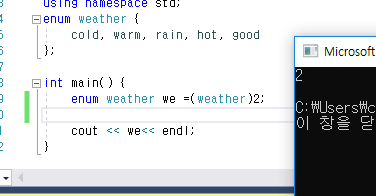
C와 마찬가지로 한 개 이상은 올 수 없다.

- 열거 타입이 정수로 강제 변환 가능한가?



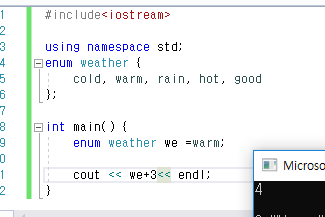
정수로 강제 변환이 가능하다.

- 다른 타입이 열거 타입으로 강제 변환 가능한가?



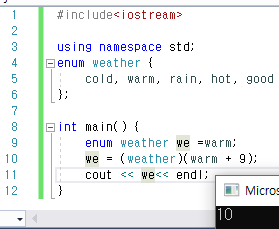
2가 출력되는 것으로 보아 가능 하다

- 열거 타입에 대한 산술 연산이 가능한가?

. 

4가 출력 된것으로 보아 가능하다.

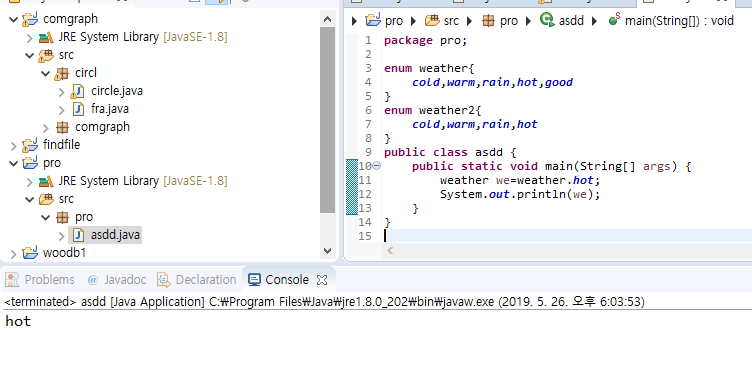
- 열거 타입 변수에 범위를 벗어난 값을 배정 가능한가?



0에서 4까지의 값을 가지는데 10이 출력된 것으로 보아 배정 가능하다.

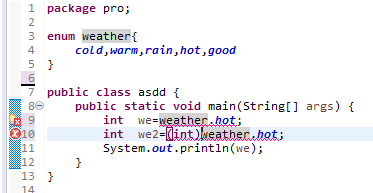
**JAVA**

- 한 열거 상수가 한 개 이상의 타입 정의에 올 수 있는가?



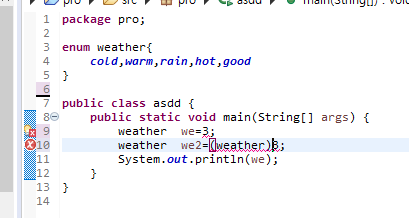
한 개 이상의 타입정의에 올 수 있다

- 열거 타입이 정수로 강제 변환 가능한가?



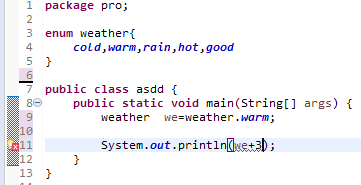
오류가 나는 것으로 보아 불가능 하다는 것을 알 수 있다

- 다른 타입이 열거 타입으로 강제 변환 가능한가?



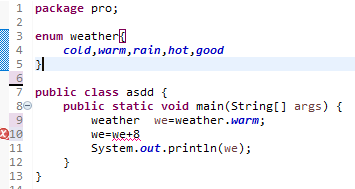
오류 메세지가 뜨는 것으로 보아 불가능 하다.

- 열거 타입에 대한 산술 연산이 가능한가?



오류 메시지가 뜨는 것으로 보아 불가능 하다.

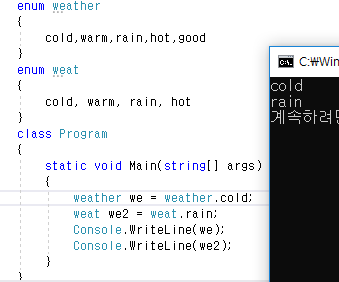
- 열거 타입 변수에 범위를 벗어난 값을 배정 가능한가?



오류 메시지가 뜨는 것으로 보아 불가능 하다.

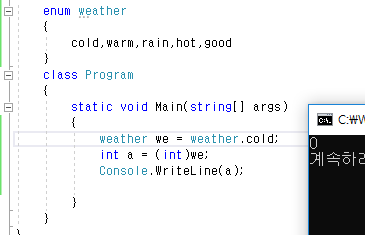
**C#**

- 한 열거 상수가 한 개 이상의 타입 정의에 올 수 있는가?



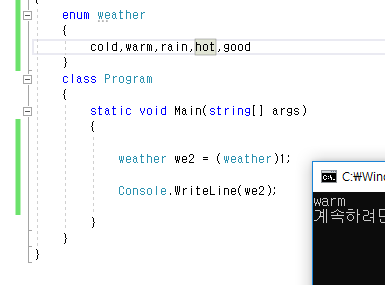
열거상수를 중복 정의해도 잘 되는 것으로 보아 한 개 이상의 타입 정의에 올 수 있다.

- 열거 타입이 정수로 강제 변환 가능한가?



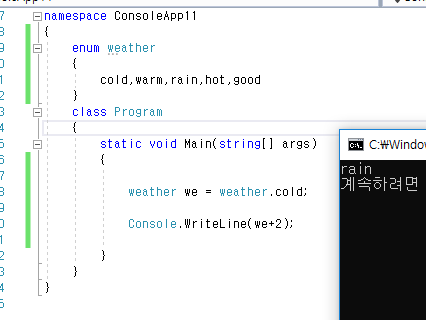
잘 작동하는 것으로 보아 강제 변환이 가능하다.

- 다른 타입이 열거 타입으로 강제 변환 가능한가?



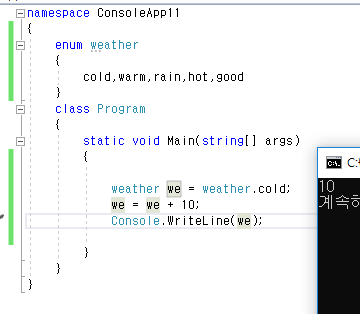
1에 해당하는 warm이 출력되는 것으로 보아 잘 작동한다.

- 열거 타입에 대한 산술 연산이 가능한가?



2를 더해줬더니 rain이 나온 것으로 보아 산술 연산이 가능하다.

- 열거 타입 변수에 범위를 벗어난 값을 배정 가능한가?



벗어난 범위지만 10이 출력되는 것으로 보아 가능하다.

**6장 연습 문제**

**3. 부동 소수점 타입에 의해서 표현될 수 있는 값들의 모임은 정확도와 범위의 관점에서 정의된다. 정확도와 범위란 무엇인가? 단정도와 배정도 표현을 위한 IEEE 부동 소수점 표준754 형식을 보여라.**

정밀도는 값의 소수점 이하 부분에 대한 정확도이며 비트의 개수로 측정된다. 범위는 소수점 이하 부분에 대해서 적어도 두 배 정도의 비트 개수를 제공한다

**단정도**

|  |  |  |
| --- | --- | --- |
| **부호 비트** | **지수(8비트)** | **소수부(23비트)** |

**배정도.**

|  |  |  |
| --- | --- | --- |
| **부호 비트** | **지수(11비트)** | **소수부(52비트)** |

**7. C++의 포인터와 참조 타입 변수를 비교하라.(행렬 곱셈을 예제로 사용하라)**

|  |
| --- |
| 전체적인 소스 |
|  |
| 매개 변수가 포인터 일 때와 참조타입 일 때를 비교해 보았다. |
| **포인터 일 때** |
|  |
| **참조타입 일 때** |
|  |
| C++에서는 매개변수는 값-호출로 전달된다. 포인터를 매개변수로 전달하면 동일한 양방향 통신 효과를 가지나, 포인터의 형식 매개변수는 명시적 역참조를 요구하며, 이는 코드를 덜 판독적이고 덜 안전하게 한다. 참조 매개변수는 다른 매개변수와 똑같이 피호출 함수에서 참조된다. 호출한수는 그 대응된 형식 매개변수가 참조 타입인 매개변수가 특별하다고 명세할 필요가 없다. 컴파일러는 참조매개변수에 값이 아닌 주소를 전달한다. |