1. Languages that employ a reference model of variables also tend to employ automatic garbage collection. Is this more than a coincidence? Explain.
2. Consider the following expression in C: a/b > 0 && b/a > 0. What will be the result of evaluating this expression when a is zero? What will be the result when b is zero? Would it make sense to try to design a language in which this expression is guaranteed to evaluate to false when either a or b (but not both) is zero? Explain your answer.
3. Why is it generally expensive to catch all uses of uninitialized variables at run time?
4. 그렇습니다, 왜냐하면 reference model에는 동일한 객체의 복사본이 있을 것이기 때문입니다.

그들 모두를 수동으로 접근(추적)하는 것은 매우 시간이 오래걸리는(지루한) 일입니다.

1. 0/b = 0, 0 > 0이므로, (0 &&어느것) = 0이기 때문에 즉시 short circuit이 발생할 수 있다.

b가 0일 때, a/b는 0으로 나누기 때문에 컴파일러는 부동 소수점 예외를 부여한다.

아니다. 0으로 나누어진 것을 피하려면 a 또는 b가 0인지에 따라 좌우에 대한 evaluating 과 short circuiting에서 전환해야 할 것이다. 그러나 어느 쪽이 0 대 부동 소수점 오차를 초래하느냐에 따라 어느 쪽이 먼저 evaluate되는지를 전환할 수 있는 신뢰할 수 있는 방법은 없다.

|  |
| --- |
|  |
|  |

1. 왜냐하면 메모리에 있는 변수 표현의 가능한 모든 비트 패턴이 어느 정도 정당한 값을 지정한다면, 초기화된/초기화되지 않은 플래그를 보관하기 위해 별도의 여분의 공간을 어딘가에 할당해야 하기 때문이다.