6장 연습문제

1. 정적 타입 언어와 동적 타입 언어에 대해서 설명하시오. 각각의 장단점에 대해서 논하시오.
205,206쪽 참조

2. 다음 Python의 예를 실행하고 결과를 타입 관점에서 설명하시오.

```
>> def add(a,b):
       return a+b

>> add(5, 10)
15                                  //정수 덧셈

>> add(5, 10.0)
15.0                        //정수5를5.0으로 자동 형변환하여 실수 덧셈

>> add(5.0, 10.0)
15.0                        //실수 덧셈

>> add(5, “10”)              //정수와 문자열 연산 불가로 인한 오류

>> add(“hello”, “world”)
'helloworld'                //문자열 접합

>> add([1,3,5], [7,9])
[1, 3, 5, 7, 9]             //리스트 병합
```

3. 다음과 같이 선언된 2차원 배열을 열 우선 방식으로 메모리에 배치한다고 할 때 배열의 원소인a[i][j]의 주소는 무엇인가? base를 이 배열의 시작주소라고 가정하자.

```
int a[m][n];
```

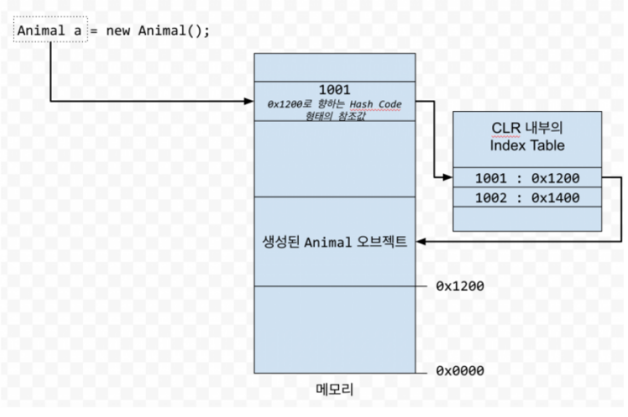
a[i][j]의 주소= base + (i + j*m)*sizeof(int)

4. 강한 타입 언어와 약한 타입 언어에 대해서 설명하시오. 각각의 장단점에 대해서 논하시오.
206,207쪽 참조

5. 포인터 타입과 참조 타입에 대해서 설명하시오. 그들의 차이점과 공통점은 무엇인가?

참조 타입과 포인터 타입은 모두 메모리 상의 객체에 대한 참조를 저장하는 데 사용되는 데이터 타입입니다. 하지만 중요한 차이점이 있습니다. 포인터는 주소를 직접 저장하지만, 참조는 해당 객체를 가리킵니다. 포인터는 연산자를 사용하여 객체를 간접 참조할 수 있지만, 참조는 자동 주소참조를 통한 직접 참조가 이루어지므로 이러한 연산자를 필요로 하지 않습니다. 즉 포인터는 간접 참조 연산자 "*"를 사용하여 객체를 참조할 수 있습니다. 반면 참조는 해당 객체를 자동 주소참조하기 때문에 이러한 연산자를 필요로 하지 않습니다.

참고



7장 연습문제

4. 다음 함수 정의를 타입 검사하고 그 타입 검사 과정을 설명하시오.

```
fun int f(int x)
  x = x - 1;
  if (x > 0) then return x;
  else return -x
}
```

이 함수 정의는int -> int타입이다. 이에 대한 타입 검사 과정은 다음과 같다.
타입 환경T = {x -> int}하에서

(1) x = x–1은 타입이 올바르게 사용되고 결과는void타입
(2) if문은then부분과else부분 모두int타입이므로int타입이다.
결과적으로 이 함수 본체는int타입이다.
(3)따라서 이 함수 정의는int -> int타입이다.

2.

5. 단형(monomorphic)타입 시스템과 다형(polymorphic)타입 시스템에 대해서 조사하여 비교 설명하시오.

X

단형(monomorphic)타입 시스템과 다형(polymorphic)타입 시스템은 프로그래밍 언어에서 타입을 다루는 방식에 대한 차이입니다. 단형 타입 시스템은 모든 변수와 표현식이 하나의 고정된 타입을 가진다는 것을 의미합니다.이는 변수가 정의될 때 타입이 결정되며, 그 이후에는 변경될 수 없다는 것을 의미합니다.대부분의C나Pascal과 같은 언어에서 사용되는 방식입니다. 예를 들어, C언어에서는 다음과 같은 정수형 변수를 선언할 수 있습니다.

```
int x = 5;
```

이 경우, x변수는 정수형(int)이며, 이후에는 다른 타입의 값을 저장할 수 없습니다.

다형(polymorphic)타입 시스템은 하나의 함수나 클래스가 여러 가지 타입을 처리할 수 있는 기능을 제공하는 타입 시스템입니다. 즉, 동일한 함수나 메서드를 다른 타입의 인자를 받아 처리할 수 있도록 타입을 일반화시켜 사용하는 것을 말합니다.

대표적으로 제네릭(generics)이 다형 타입 시스템의 한 예입니다. 제네릭은 하나의 메서드나 클래스를 선언할 때 타입 파라미터(type parameter)를 사용하여 일반화시킵니다. 이렇게 일반화된 메서드나 클래스는 다양한 타입의 인자를 받아들일 수 있으며, 코드의 재사용성을 높이는 효과를 가져옵니다.

다형 타입 시스템은 코드의 재사용성을 높여주는 장점뿐만 아니라, 타입 안정성을 높일 수 있는 효과도 있습니다. 일반화된 코드를 사용함으로써, 타입 변환에 따른 오류를 사전에 방지할 수 있으며,코드의 가독성도 향상시킬 수 있습니다.