Chapter 4 변수

Property	
날짜	@October 13, 2021
발표자	김윤서

4.1 변수란?

변수 : 값을 저장하는 메모리 공간 → 쉽고 효과적으로 데이터 조작

4.2 변수 선언

변수 선언 : 메모리 할당 명령

```
var a int = 10
```

- var : 변수 선언 키워드
- a : 변수명
- int : 변수 타입
- = 10 : 초깃값

4.3 변수에 대해 더 알아보기

4.3.1 변수의 4가지 속성

- 이름 : 메모리 공간에 쉽게 접근
- 값 : 메모리 공간에 저장된 값
- 주소 : 메모리 공간의 시작 주소
- 타입 : 자료형

4.3.2 변수는 이름을 가지고 있다

변수명 규칙

- 변수명은 문자, _, 숫자 사용 가능. 단, 첫 글자는 반드시 문자나 _로 시작해야함
- _를 제외한 다른 특수문자를 포함할 수 없음

권장 사항

- 변수명은 영문자 제외 다른 언어의 문자 사용하지 않음
- 카멜케이스를 따른다 (두번째 단어부터는 대문자로 시작)
- 되도록 짧게
- 밑줄은 일반적으로 사용하지 않음

4.3.3 변수는 타입을 가지고 있다

왜 필요한가?

1. 공간 크기를 나타낸다 : 메모리 시작 주소 → 크기를 알아야함
2. 컴퓨터가 데이터를 해석할 수 있다

숫자 타입

Aa 이름	설명	값의 범위
uint8	1바이트 부호 없는 정수	0 ~ 255
uint16	2바이트 부호 없는 정수	0 ~ 65535
uint32	4바이트 부호 없는 정수	0 ~ 4294967295
uint64	8바이트 부호 없는 정수	0 ~ 18446744073709551615
int8	1바이트 부호 있는 정수	-128 ~ 127
int16	2바이트 부호 있는 정수	-32768 ~ 32767
int32	4바이트 부호 있는 정수	-2147483648 ~ 2147483647
int64	8바이트 부호 있는 정수	-9223372036854775808 ~ 9223372036854775807
float32	4바이트 실수	IEEE-754 32비트 실수
float64	8바이트 실수	IEEE-754 64비트 실수
complex64	8바이트 복소수(진수, 가수)	

Aa 이름	설명	값의 범위
complex128	16바이트 복소수(진수, 가수)	
byte	uint8의 별칭	
rune	int32의 별칭	
int	32비트 컴퓨터에서는 int32, 64비트 컴퓨터에서는 int64	
uint	32비트 컴퓨터에서는 uint32, 64비트 컴퓨터에서는 uint64	

그외 타입

- 불리언
- 문자열
- 배열
- **슬라이스** : 가변 길이 배열
- 구조체
- 포인터
- 함수 타입
- 인터페이스
- 맵
- **채널** : 멀티스레드 환경에 특화된 큐 형태 자료구조

4.4 변수 선언의 다른 형태

```
package main

import "fmt"

func main() {
  var a int = 3 // 가장 기본 형태
  var b int     // 초깃값 생략 - 초깃값은 타입별 기본값으로 대체
  var c = 4     // 타입 생략 - 변수의 타입은 우변 값의 타입이 됩니다
  d := 5        // 선언대입문 := 을 사용해서 var 키워드와 타입 생략

  fmt.Println(a, b, c, d)
}
```

숫자값 기본 타입 : 정수 - int, 실수 - float64

선언 대입문 := : var 키워드, 타입 생략 가능

4.5 타입 변환

```
package main

import "fmt"

func main() {
  a := 3              // int
  var b float64 = 3.5 // float64

  var c int = int(b)  // float64에서 int로 변환
  d := float64(a * c) // int에서 float64로 변환

  var e int64 = 7
  f := int64(d) * e // float64에서 int64로 변환

  var g int = int(b * 3) // float64에서 int로 변환
  var h int = int(b) * 3 // float64에서 int로 변환 g와 값이 다릅니다.
  fmt.Println(g, h, f)
}
```

같은 숫자값이라도 타입이 다르면 연산이 안 된다! → 타입 변환

유의점

1. 실수 타입 → 정수 타입 : 소수점 이하 삭제
2. 큰 범위 → 작은 범위 : 값이 변할 수도 있음

예)

```
var a int16 = 3456
var c int8 = int8(a) // -128
```

4.6 변수의 범위

```
//ch4/ex4.6/ex4.6.go
package main

import "fmt"
```

```
var g int = 10 // 패키지 전역 변수 선언 ❶

func main() {
  var m int = 20 // 로컬 변수 선언 ❷

  {
    var s int = 50 // 로컬 변수 선언 ❸
    fmt.Println(m, s, g)
  } // s 로컬변수는 사라짐 - ❹

  m = s + 20 // Error - ❺
} // main함수 끝 - ❻
```

4.7 숫자 표현

4.7.1 정수 표현

부호 있는 정수 타입

- 첫 번째 비트 = 부호 비트 (1 : 음수, 0 : 양수)
- 보수로 표현

예) -15 ⇒ 절댓값 0 000 0000 0000 0111 ⇒ 비트 반전 ⇒ 1 111 1111 1111 0000 ⇒ +1 ⇒ 1 111 1111 1111 0001

4.7.2 실수 표현

IEEE-754 표준

예) 1024.234 = 0.1024234 x 10^4 = 0.1024234e+04

4바이트 실수 : 부호비트 (1) + 지수부 (8) + 소수부 (23)

float32 : 7자리

float64 : 15자리

Go 언어에서 실수 표현 : 정확한 값이 아닌 타입이 허용하는 범위에서 **가장 가까운 근삿값**으로 표현

```
//ch4/ex4.7/ex4.7.go
package main

import "fmt"

func main() {
  var a float32 = 1234.523
```

```
  var b float32 = 3456.123
  var c float32 = a * b
  var d float32 = c * 3

  fmt.Println(a)
  fmt.Println(b)
  fmt.Println(c)
  fmt.Println(d)
}
```

```
1234.523
3456.123
4.266663e+06
1.2799989e+07
```

정확한 수치 연산이 필요한 프로그래밍에서는 각별히 주의!!

연습문제

1. a : int, b : float64, c : string, d : int32, e : float32
2. 360은 int8의 범위(-128~127)을 초과한다. 그래서 타입 변환하면서 상위 3바이트가 삭제되었을 때 값이 변하였다.
3. f1은 우변의 결과가 float32의 자릿수 제한으로 값이 42707.406으로 변하였다. f2는 123.546789가 float32로 형변환이 되어 123.54679가 되었고 곱한 결과는 근삿값으로 변환되어 자릿수 탈락으로 42707.41로 변하였다. (자릿수 헷갈림 주의!)