Variables and Types

Created	@Jan 22, 2021 4:00 PM
Tags	Lesson

Literals

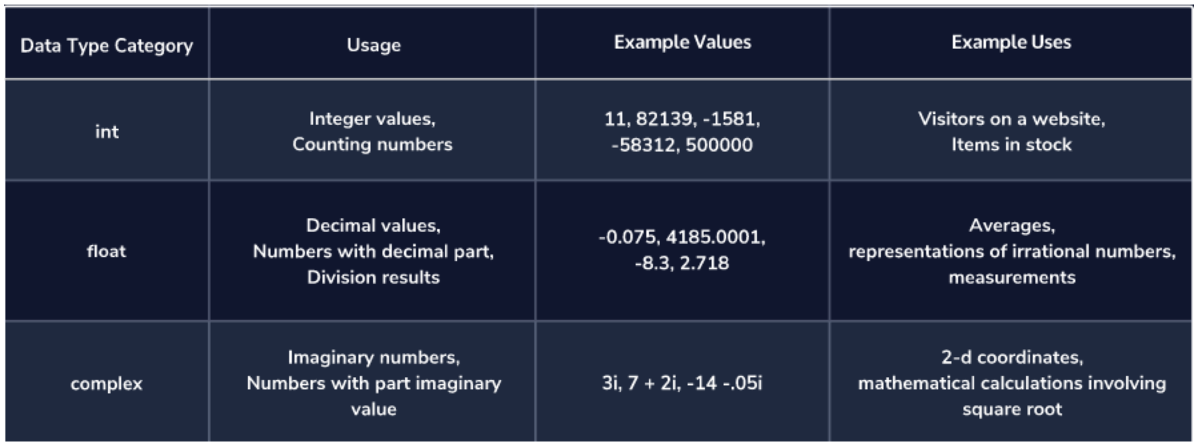
- Numbers, Strings, etc. that are unnamed and written into code directly
- Standard arithmetic operations:
    - `+` to add
    - `-` to subtract
    - `*` to multiply
    - `/` to divide (defaults to integer division)
    - `%` modulo
- `fmt.Println(2235 * 1231)`

Constants

- `const` denotes an unchanging named value
- `cost earthsGravity = 9.80665`
  `fmt.Println(earthsGravity)`

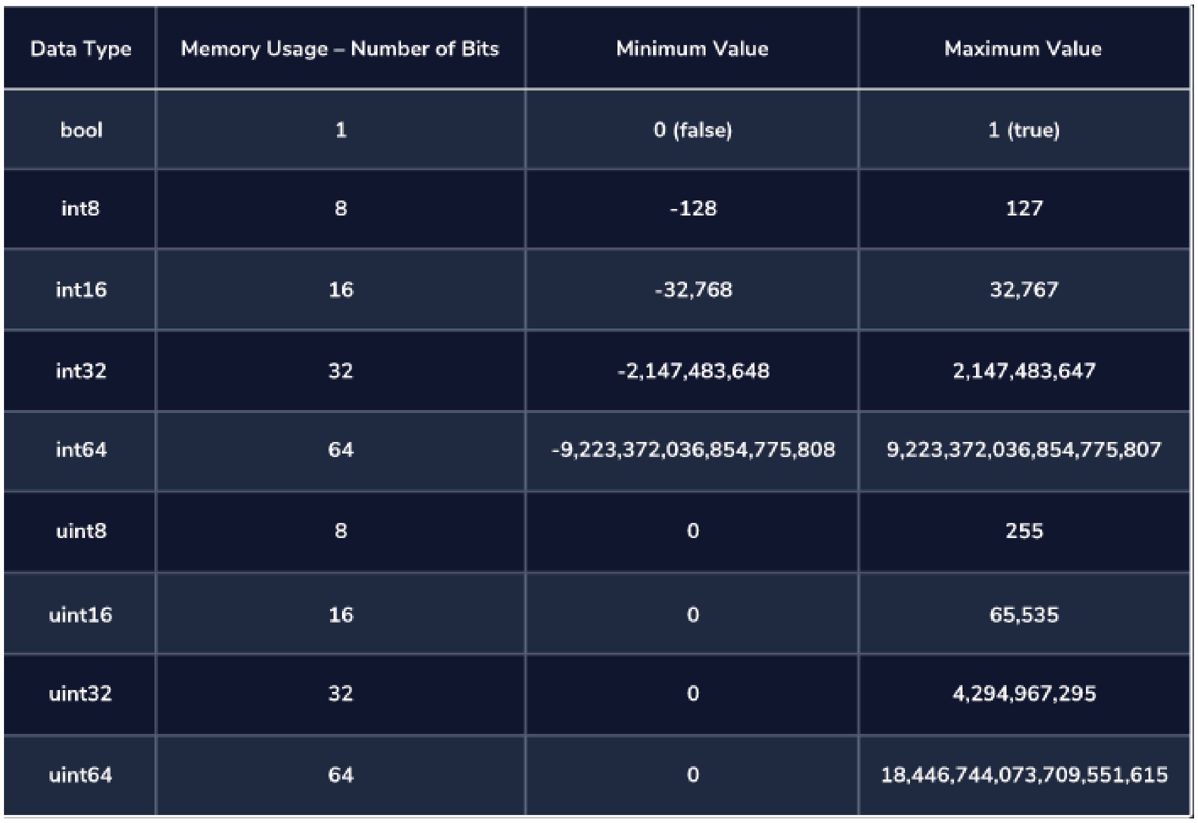
Data Types

- Basic Number Categories



Data Type Category	Usage	Example Values	Example Uses
int	Integer values, Counting numbers	11, 82139, -1581, -58312, 500000	Visitors on a website, Items in stock
float	Decimal values, Numbers with decimal part, Division results	-0.075, 4185.0001, -8.3, 2.718	Averages, representations of irrational numbers, measurements
complex	Imaginary numbers, Numbers with part imaginary value	3i, 7 + 2i, -14 -.05i	2-d coordinates, mathematical calculations involving square root

- Specific Number datatypes



Data Type	Memory Usage – Number of Bits	Minimum Value	Maximum Value
bool	1	0 (false)	1 (true)
int8	8	-128	127
int16	16	-32,768	32,767
int32	32	-2,147,483,648	2,147,483,647
int64	64	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
uint8	8	0	255
uint16	16	0	65,535
uint32	32	0	4,294,967,295
uint64	64	0	18,446,744,073,709,551,615

  - Go also has two floating point types `float32` and `float64` and two complex number types

Variables

- Variables are defined by the `var` keyword

```
var lengthOfSong uint16
var isMusicOver bool
var songRating float32
```

- The above code creates three variables
    - An unsigned 16-bit integer called `lengthOfSong`
    - A Boolean value called `isMusicOver`
    - A 32-bit float called `songRating`

Reading Errors

- Go tries to tell you what the issue is by offering you the following pieces of information:
    - the name of the file where something went wrong
    - the line number in that file where Go noticed an issue,
    - the column number (the number of characters from the left) on that line where the error occurred.
- Go throws an error when a variable is declared that is not used:

```
package main

func main() {
  var numberWheels int8
}
```

- The above code throws error `./main.go:4:7: numberWheels declared and not used`

Assigning Variables

- Variables are assigned values with the `=` assignment operator
- Variables can be *initialized* at declaration, or afterwards

```
var kilometersToMars int32
kilometersToMars = 62100000
```

```
var kilometersToMars int32 = 62100000
```

- Default value of numeric variables is `0`

Strings

- Strings are defined with `"` double-quotes (`'` single quotes have another meaning)
- Strings can be concatenated with `+` without any additional spaces or punctuation
- `var description string` defines a string variable named description
- Default value is `""`

Inferring Type

- The short declaration operator `:=` can be used in place of a variable type
- The following declarations are equivalent

`nuclearMeltdownOccurring := true`	`var nuclearMeltdownOccurring = true`
`radiumInGroundWater := 4.521`	`var radiumInGroundWater = 4.521`
`daysSinceLastCatastrophe := 0`	`var daysSinceLastCatastrophe = 0`
`externalMessage := "Everything is normal. Keep calm and carry on."`	`var externalMessage = "Everything is normal. Keep calm and carry on."`

Updating Values

- Values can be updated with special shorthand operators
  - `+=` for addition
  - `-=` for subtraction
  - `*=` for multiplication
  - `/=` for division
  - `%=` for modulo

Declaring Multiple Variables

- `var part1, part2 string`
- `quote, fact := "Bears, Beets, Battlestar Galactica", true`

```go
package main

import "fmt"

func main() {
  var congrats string
  congrats = "Congratulations"
  congrats += "!!!"
  fmt.Println(congrats)

  var challenge string = "What else can you do?"
  fmt.Println(challenge)

  reminder := "Pratice is important!"
  fmt.Println(reminder)
}
```

Quiz Scorecard
100%
👏 Great job!
✗ 0
✓ 9
Up Next
Retake Quiz