Intro to Golang

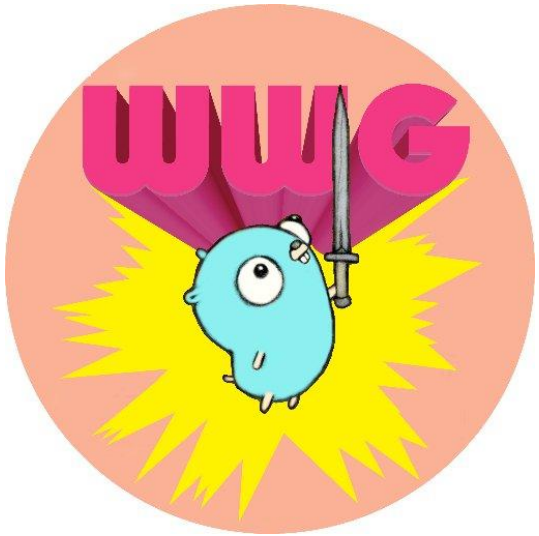
Women Who Go & Girl Develop It

July 14th, 2018

Women Who Go

Women Who Go is an inclusive space for women, non-binary and transgender people with interest in the Go programming language, or programming in general. Women Who Go works to build a more diverse and inclusive golang community.

@womenwhogo_nyc



Girl Develop It

Want to learn how to code? Have a great idea? Don't be shy. Develop it. Girl Develop It is an non-profit organization that exists to provide affordable and accessible programs to women who want to learn software and web development through mentorship and hands-on instruction.

@gdinyc

Class Goals

Review language-agnostic concepts like variables, conditionals, looping, etc and demonstrates their use with the Go language syntax. Cover only thefundamentals, avoid introducing some of the more advanced capabilities of Go (e.g. goroutines, channels, etc).

- Working with variables and constants
- Working with conditionals, if/else statements
- Working with lists and looping
- Using functions for reusing functionality
- Using complex types to represent concepts out in the real world

Class Agenda

10:30 - 10:45: Introduction to workshop and class

10:45 - 11:30: Values, variables, constants, for loops, if/else, switch

11:30 - 11:45: **Break**

11:45 - 12:30: Arrays and Slices

12:30 - 1:30: **Lunch**

1:30 - 2:30: Map, Range, Functions

2:30 - 2:45: **Break**

2:45 - 3:15: Project 1

3:15 - 3:30: Future Learning and Wrap Up

10:45 - 11:30

Lesson 1 - 7

Let's Get Started!

Lessons:

https://jboursiquot.gitbooks.io/learn-to-code-with-go-workshop/

Go by Example:

https://gobyexample.com/hello-world

Go Playground:

https://play.golang.org/

Lesson 1

Challenge

Rewrite your program so that you only make use of a single print statement while still outputting your story on multiple lines.

fmt.Println automatically puts a newline at the end of your string. Alter your code to make use of fmt.Print instead while still displaying your story on multiple lines when your program is executed.

Advanced Challenge

Install Go on your local computer. Go can be downloaded at https://golang.org/dl/.

Install our editor of choice: Visual Studio Code. (Or use your preferred editor)

Copy your simple program from the Challenge above and run it from your editor.

Lesson 2

Challenge

Get your program to print out the following sentence:

Hi, I'm NAME and I am AGE years old. My favorite color is COLOR and I have $X.XX in my pocket!

Each time there's something in CAPITALS, use a comma and use the right Go type for that thing. For example:

fmt.Println("My favorite color is ", "blue")

Advanced Challenge

Do the challenge again, this time, using fmt.Printf and the appropriate format verbs.

Use the appropriate format verb to print the the string The integer value for the letter a is 97

Lesson 3

Challenge

Variables are perfect for storing data that can be used later on within a program.

Update the challenge from Lesson 2 to use variables for the following values name, age, color, money

Advanced Challenge

In Go, it is common to assign the results of a function to one or more variables. For this challenge update the code from above to use fmt.Sprintf which will return a string version of the formatted output, and store the return value to a new variable named msg

Lesson 4

Challenge

Go has many constants already defined, for example `math.Pi`.

Print out the perimeter distance around a circle that has a radius of 5 feet.

The formula for perimeter distance is `2 * math.Pi * radius`.

Advanced Challenge

Update the code above to also output the area of the same circle. The formula for area is `math.Pi * radius^2`

Lesson 5

Challenge

Write a for loop that only executes once

Write a for loop with no condition statement that exits when the count is equal to 100.

Advanced Challenge

Use the for loop to write a program that outputs a right isosceles triangle of height and width n, so n = 3

*

***

*****

Lesson 6

Challenge

Combine a For loop with an If/Else to count to 10 and print out if a number is even or odd.

Advanced Challenge

Combine a for loop with if, else if and else to count to 10 and print whether the number is less than 5, equal to 5, or greater than 5. The output should look like:

Lesson 7

Challenge

Combine a for loop with switch to count to 10 and print whether the number is less than 5, equal to 5, or greater than 5. The output should look like:

How many different ways can you write your switch cases?

11:30 - 11:45
Break

11:45 - 12:30

Lesson 8 - 9

Lesson 8

Challenge

Write a program that does the following:

- Creates an array of size 10 and print it out -> [0 0 0 0 0 0 0 0 0 0]
- Loops through the numbers 1 through 10, and set the value in each slot of the array to the number in the loop
- Prints out the array [1 2 3 4 5 6 7 8 9 10]
- Creates a boolean array of size 10
- Loops through the first array, and set the second array's value to true if the number in the first array is even
- Prints out the second array -> [false true false true false true false true false true]

Advanced Challenge

Declare an int array of size 5 that is initialized as [1 0 0 3 0]

Lesson 9

Challenge

Write a program that does the following:

- Creates an empty slice of integers and prints it out -> []
- Loops over the numbers 1 through 10 and appends them to the slice
- Prints out the slice -> [1 2 3 4 5 6 7 8 9 10]
- Prints out the first 5 numbers in the slice -> [1 2 3 4 5]

Arrays vs Slices

Array

- Length is part of it's type
- Cannot be resized
- Do not need to be initialized explicitly; the zero value of an array is a ready-to-use array whose elements are themselves zeroed
- Arrays are values (as opposed to pointers)

Slices

- Types described by the element they contain
- Have helpful methods to enable resize like operations, like "append" and "slice"
- Need to be explicitly initialized with the "make" operation
- Slices are pointers to underlying arrays and the length and capacity of the slice

12:30-1:30
Lunch

1:30-2:30

Lesson 10 - 12

Lesson 10

Challenge

Write a program that does the following:

- Creates a map from integers to booleans
- Loops through the numbers 1 through 10 and sets the map where the key is the number and the value is whether the number is even (HINT: use an If/Else or Switch!)
- Prints out the map -> map[1:false 2:true 3:false 4:true 5:false 6:true 7:false 8:true 9:false 10:true]
- Prints out whether 4 is even using variables -> The number 4 is even: true

Advanced Challenge

Write a program that sorts a set of numbers 0 - 30 into two groups: "Numbers divisible by 9", "Numbers not divisible by 9". (Hint: use constants as map keys.) The output for the program should something like the output below:

- Divisible by 9: [9 18 27]
- Not Divisible by 9: [10 11 12 13 14 15 16 17 19 20 21 22 23 24 25 26 28 29 30]

Lesson 11

Challenge

Modify your program from lesson 10 to use a range to iterate the map to output the following:

1 is odd 2 is even 3 is odd 4 is even 5 is odd 6 is even 7 is odd 8 is even 9 is odd 10 is even

HINT: use a for loop with range to get the key and value of each entry in the map, then use an If/Else on the value.

Advanced Challenge

When iterating over maps the order of keys is not guaranteed, so running the same program multiple times will result in different outputs, as you may have noticed from the challenged above. To print a map in order the keys must be sorted separately and then used to access the fields of the key.

Update the program above so that the map is always printed in order starting at 1, all they way to 10.

HINT: use the sort.Ints function in the sort package

Lesson 12

Challenge

Write a program that does the following:

- Loop through the numbers 1 through 10
- Call a function that prints out if the number is even or odd (like in Lesson 11's output).
- Refactor your program to add a new function that takes an integer and returns a boolean signifying if the integer is even, use that function inside your printing function.

Advanced Challenge

Refactor your code so that you end up with a third function. Extra bonus points if you have a function in a function in a function!

2:30-2:45
Break

2:45-3:15

Project 1

Project

We covered a lot of ground with things like variables, conditionals, and looping. In order to help strengthen those concepts we will write a quiz program that displays (prompts) a question to the user and waits for the user to enter an input (response) to the question being asked. Once all questions have been answered the program should display the total number of questions the user got correct, over the total number of questions asked (i.e You got 5/6 correct).

Hint: You will need a type that can hold both the questions and answers as pairs, and the fmt.Scanln function from the fmt package for reading input from the screen.

- What is 5+5? 10
- What is 1+1? 2
- What is 8+3? 12
- What is 1+2? 3
- What is 8+6? 14
- You got 4/5 correct!

Next Lessons & Future Learning

Pointers

- Allow you to pass references to values and records within your program.

Structs

- Go's structs are typed collections of fields. They're useful for grouping data together to form records.

Methods

- Go does not have classes. However, you can define methods on types.
- A method is a function with a special receiver argument.

Next Lessons & Future Learning

Go Basics and Beyond

Concurrency Patterns:
https://blog.golang.org/advanced-go-concurrency-patterns

Channels:
https://www.youtube.com/watch?v=KBZlN0izei

Composition:
https://www.ardanlabs.com/blog/2015/09/composition-with-go.html