Go Track
Week 02 Practical

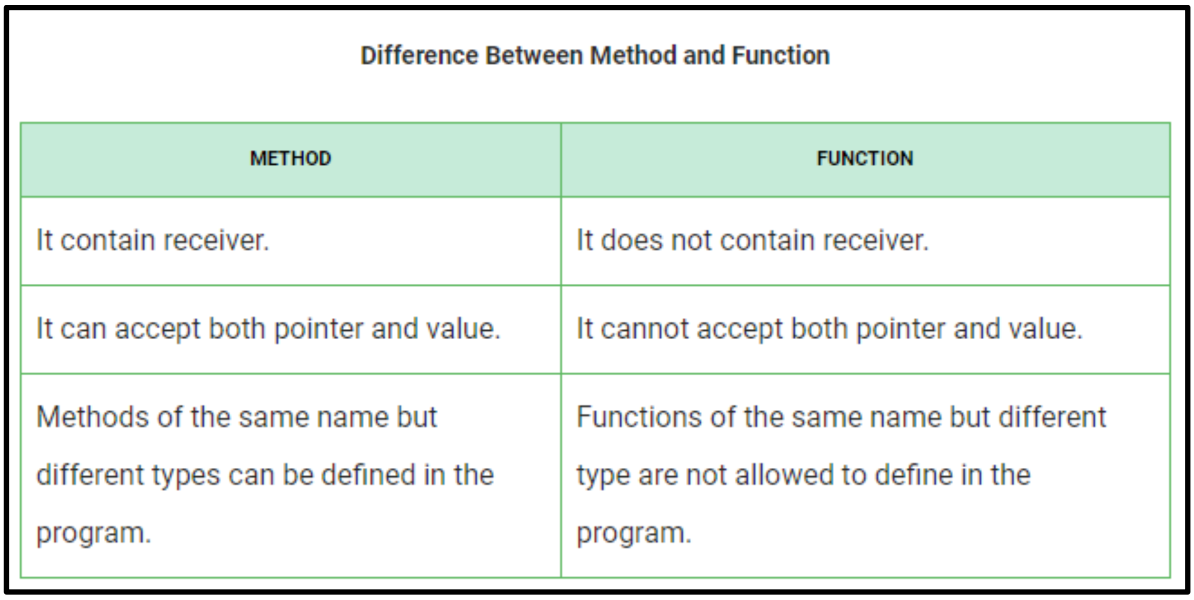
Function and Methods

Description	***In this lesson, you will learn the basics of functions and methods***
Learning Objectives	• **Know how to create simple functions** • **Know recursive functions** • **Know how to create and access methods** • **Know how to use multiple file functions and methods**
Duration	**20 minutes**

Functions and Methods

It is a collection of statements that perform some specific task. It can either return the result or return nothing. A method contains the receiver argument to allow it to access the properties of the receiver.

Struct type receiver	**The receiver is defined from a struct**
Non-Struct type receiver	**The receiver is defined on a non-struct type**
Pointer receiver	**The receiver is defined with a pointer type**



Difference Between Method and Function

METHOD	FUNCTION
It contain receiver.	It does not contain receiver.
It can accept both pointer and value.	It cannot accept both pointer and value.
Methods of the same name but different types can be defined in the program.	Functions of the same name but different type are not allowed to define in the program.

Source:
https://www.geeksforgeeks.org/methods-in-golang/

Activity #1 Factorial Function

Create a simple factorial calculator using a recursive function.
Request for the user to enter a number.
Print out the factorial value of the number and print to the user.

Activity #2 Customer Methods

Create a file customer.go to contain:
A customer Struct with the following information

1. First Name, string
2. Last Name, string
3. Username, string
4. Password, string
5. Email, string
6. Phone, int
7. Address, string

Create the following methods for customer

1. Retrieve userCredentials for password and username
2. Retrieve userAddress
3. Print allUserInformation

Create a file main.go to contain:
A customer with the following

1. Variable, Customer1
2. First Name, “Micheal”
3. Last Name, “Jordan”
4. Username, “MJ2020”
5. Password, “1234567”
6. Email, “MJ2020@gmail.com”
7. Phone. 12345678
8. Address, “18227 Capstan Greens Road Cornelius, NC 28031.”

Print out all the info of the customer in main.go
Print out the retrieved value of user credentials and user address in main.go

Activity #3 The "Games" Shop

Create an application that would list out the different games available and their corresponding prices for each game.

Create a file main.go,
1. To create the different game objects and pass them to the list to print.
2. To arrange the games into a list and pass to the list.go to show.

Create a file game.go,
1. To contain the struct definition of game.
   a. title , string
   b. price, float64
2. To contain the method to print the price of each game.

Create a file list.go,
1. To receive the list content and print out the content.

The games should be
1. Minecraft , $5
2. World of warcraft, $19
3. Elite Dangerous, $54

NGEE ANN POLYTECHNIC

Interface

Description	*In this lesson, you will learn the basics of Interface*
Learning Objectives	• Know how to create basic interface • Further appreciate the use of interface for polymorphism
Duration	20 minutes

Interface

It is a custom type that is used to specify a set of one or more method signatures.

ⓘ	**Only in Go**	**It is defined differently from other languages.**
⚠	**It is not allowed to create an instance of the interface.**	

Activity #1 Interface

Use the game shop application created under "Function and Methods" activity #3.

Modify the application to include a new file for books.go that would contain the following struct for book

1. Title , string
2. price, int

Modify the application to include a new file for computerAccessories.go that would contain the following struct for computerAccessories

1. Title, string
2. price, int

Modify the application to include the use of interface so that the new books and computer accessories can be included in the list.

The books should be

1. Candle in the tomb, $20
2. Barney and Friends, $10

The computer accessories should be

1. Razer BT earpiece, $159
2. Razer keyboard, $110
3. Logitech Mouse, $80

Print the new list along with the existing items.

Reflection

Description	*In this lesson, you will learn the basics of Interface*
Learning Objectives	• Know how to apply basic inspection on types
Duration	20 minutes

Reflection

It is the ability for a program to inspect and analyse its structure during run-time.

ⓘ	**TypeOf**	**Used to determine the type**
	ValueOf	**Used to determine the value**
	Kind	**Used to determine the kind**
	NumField	**Used for structs to determine the number of fields present**
	Field	**Used for structs to access each field by indexing variable**
⚠	**Using the types found during runtime allows for generic code to be written but should be used with caution**	

Activity #1 Basic Reflection

Create a simple application that has a function inspect.
The inspect function takes in an empty interface and prints out the type and value of the received variable.
Declare the following variables and pass them to the function created.

1. “This is a string”
2. 12345
3. 1.2345
4. true

Activity #2 Struct reflection

Create a simple struct for customer with the following fields

1. FName string
2. LName string
3. UserID int
4. InvoiceTotal float64

Create a function inspect that would print out the type, kind and number of fields in the struct.

The customer data are

1. First Name : “John”
2. Last Name: “Wick”
3. ID: 123123123
4. Invoice Total : 10000

NGEE ANN POLYTECHNIC

Go Documentation

Description	*In this lesson, you will learn about Go Documentation*
Learning Objectives	**Understand how to properly comment Go code, and generate documentation using Go tool**
Duration	**2 hours**

Very often, software engineers and programmers will need to work in a team when it comes to building industry software. It is pertinent that the code written by all are readable, maintainable and extensible for others to be able to follow up and/or use the libraries or subsystems that are being developed.

One of important practices that are usually required is good documentation of the code. In Go, there are several practices that are usually recommended when inserting comments in code; there are also tools like godoc that can help to convert the commentary into HTML-based documentation that can make it easily accessible to others.

Activity : Generate documentation of your code.

Make further improvements to your program that you have worked on previously in Go Security to:

- Include good comments to describe well the functions, packages, important parts of code etc.
- Use godoc tool to generate HTML-based documentation from your code comments.

NGEE ANN POLYTECHNIC

Idiomatic Go

Description	*In this lesson, you will learn about Idiomatic Go*
Learning Objectives	**Understand and apply effectively the idioms and conventions of the Go programming language**
Duration	**2 hours**

Go is created based on inspiration from several existing languages, however it does have distinctive properties that make writing of effective Go programs different from the way code is written using other programming languages. Thus, to write Go programs well, it is important to understand its idioms and conventions, such as naming, formatting, programming constructs and many more, in order to maximize the benefits that the Go language can offer.

Activity : Review and make improvements to your program, applying Idiomatic Go.

Review and make further improvements to your program that you have worked on previously in Go Documentation, to ensure that good, clear Go conventions and practices are applied.