Fakultas Ilmu Komputer Universitas Indonesia

Course Outline

CSGE 601021 Foundations of Programming 2

Semester II 2023/2024

1. General Information

Teacher:

L. Yohanes Stefanus -- yohanes@cs.ui.ac.id

Teaching Assistants:

Muhammad Oka	oka@muhammadoka.dev
Muhammad Sean Arsha Galant	m.seangalant@gmail.com
Rafi Ardiel Erinaldi	rafivct@gmail.com
Ferdinand Raja Kenedy	ferdinandrajaa@gmail.com

Objective:

This course teaches the concepts and techniques of computer programming in the contexts of Introductory Computer Science, as the continuation of **Foundations of Programming 1.** The focus is on algorithmic thinking and **multi-paradigm programming in Go**, including secure coding.

References:

1. Jon Bodner. Learning Go: An Idiomatic Approach to Real-World Go Programming. Second Edition. O'Reilly Media. 2024.

O'REILLY®

Second Edition

Learning Go

An Idiomatic Approach to Real-World Go Programming

Jon Bodner

Prerequisite:

Foundations of Programming 1

Credit:

4 sks (needs at least 12 hours per week)

Status:

Required course

Class schedule:

Monday,	08.00-09.40	@ A7.15 (Gedung Baru)
Wednesday,	10.00-11.40	@ A7.15 (Gedung Baru)
Thursday,	09.00-10.40	@ Lab A1.02 (Gedung Baru)

Course website:

https://scele.cs.ui.ac.id/course/view.php?id=3790

2. Instructional Objectives

- In compliance with the curriculum of Fasilkom.
- Students will study general programming concepts, as well as a modern multi-paradigm programming language, namely Go, which illustrates those concepts. Students will design, implement and test Go programs.
- At the end of this course when presented with a problem we expect that a student will be **able to write a Go program to solve it.**

3. Course Plan

Week	Topic	Lab	Assignment
1	Introduction to Golang (Go); Programming Environment; Compiler versus Interpreter; the Go Playground	Lab 0 *: Setting up the Go programming environment *no submission is required.	
	Predeclared Types and Declarations: The Predeclared Types, The Zero Value, Literals, Booleans, Numeric Types, Runes and Strings, Explicit Type Conversion, var Versus :=, Using const, Typed and Untyped Constants, Unused Variables, Naming Variables and Constants		
2	Composite Types: Arrays, Slices, Converting Arrays to Slices, Converting Slices to Arrays, Strings and Runes and Bytes, Maps, The comma ok Idiom, Comparing Maps, Using Maps as Sets, Structs, Anonymous Structs, Comparing and Converting Structs	Lab 1: Basics of Go	
3	Blocks, Shadows, and Control Structures: Blocks, Shadowing Variables, if, for, The Complete for Statement, The Condition-Only for Statement, The Infinite for Statement, break and continue, The for-range Statement, switch, Blank Switches, goto	Lab 2:	
4	Functions:	Lab 3	Programming

	Declaring and Calling Functions, Simulating Named and Optional Parameters, Multiple Return Values, Functions Are Values, Function Type Declarations, Anonymous Functions, Closures, Passing Functions as Parameters, Returning Functions from Functions, Defer, Call by Value		Assignment 1
5	Pointers: A Quick Pointer Primer, Pointers Indicate Mutable Parameters, Pointer Passing Performance, The Zero Value Versus No Value, Slices as Buffers, the Garbage Collector	Lab 4	
6	Types, Methods, and Interfaces (1): Types in Go, Methods, Functions Versus Methods, Type Declarations Aren’t Inheritance, iota, Embedding,	Lab 5	Programming Assignment 2
7	Types, Methods, and Interfaces (2): A Quick Lesson on Interfaces, Type Assertions and Type Switches, Implicit Interfaces, Wire, Go Isn’t Particularly Object-Oriented		
8	**Review; UTS (Midterm Exam)**	-	
9	Generics: Generics in Go, Generic Functions, Generics and Interfaces, Type Inference and Generics, Generic Data Structures, Idiomatic Go and Generics	-	

10	Errors: How to Handle Errors: The Basics, Use Strings for Simple Errors, Sentinel Errors, Errors Are Values, Wrapping Errors, Is and As, Wrapping Errors with defer, panic and recover, Getting a Stack Trace from an Error	Lab 6	Programming Assignment 3
11	Modules, Packages, and Imports: Repositories, Modules, and Packages, Using go.mod, Building Packages, Working with Modules, Module Proxy Servers	Lab 7	
12	Concurrency in Go: When to Use Concurrency, Goroutines, Channels, Understanding How Channels Behave, select	Lab 8	
13	**Concurrency Practices and Patterns**: Goroutines, for Loops, and Varying Variables, Clean Up Your Goroutines, When to Use Buffered and Unbuffered Channels, Backpressure, Turn Off a case in a select, Time Out Code, Use WaitGroups, Mutexes, Atomics	Lab 9	Programming Assignment 4
14	The Context: What Is the Context? Contexts with Deadlines, Context Cancellation, Writing Tests in Go	Lab 10	
15	Reflect, Unsafe, and Cgo: Reflection Lets You Work with Types at Runtime; Types, Kinds, and Values; Build Functions with Reflection to	-	

	Automate Repetitive Tasks;
	Reflection Can't Make Methods;
	unsafe; Using unsafe to Convert External Binary Data; Cgo Is for Integration, Not Performance
	• **Review**		

16	**Review; UAS (Final Exam)**	-	

4. Evaluation Components

Programming project (4)	20%
Lab Tutorial (10)	10%
Quiz (2 or more)	10%
Midterm Exam	20%
Final Exam	40%

5. Academic Rules

- **If you cheat on an exam, you will get a final mark of E.**

➢ The goal of assignments/projects/homework is to give you practice in mastering the course material.

➢ You must write up each problem solution or program by yourself without assistance, even if you collaborate with others to solve the problem, including the use of AI-based chatbot such as chatGPT, Google-Bard, MicroSoft Bing, etc. You are asked on your assignment hand-in to identify your collaborators. If you did not work with anyone, you should write "Collaborators: none". If you obtain a solution through research (e.g., on the world wide web), acknowledge your source, but write up the solution in your own words.

➢ Programming projects are to be done individually. Discussing problem-solving strategies with other students is encouraged, BUT *as soon as the discussion turns to Python implementation that must be done on your own.*

➢ *Under no circumstances should you share a project solution with another student. Simply showing your solution to another student almost guarantees a zero score: past experience shows that a student who asks to "look at" your solution will copy parts of it.*

- Plagiarism and other anti-intellectual behavior cannot be tolerated in any academic environment.

6. Class Rules

- **Class Attendance:**
  It is expected that you will arrive on time and attend every class. If you need to miss a class due to an emergency, it will be your responsibility to obtain missed notes and course announcements from another student.
  Students who fail are those who do not attend lectures regularly.