Day 1: Introduction to Go

Overview of Go programming language

Installing Go and setting up development environment

Basic syntax and data types in Go

Writing and running a simple "Hello, world!" program

Understanding Go's philosophy and design principles

What is the primary goal of Golang?

A. To provide a high-level programming language for web development

B. To provide a low-level systems programming language with the simplicity and safety of a high-level language

C. To provide a functional programming language with advanced features like immutability and referential transparency

D. To provide a scripting language with dynamic typing and a flexible syntax

Answer: B

Who are the creators of Golang?

A. Google

B. Microsoft

C. Apple

D. IBM

Answer: A

What year was Golang released to the public?

A. 2007

B. 2009

C. 2011

D. 2013

Answer: B

Which of the following is a design goal of Golang?

A. High performance

B. Simplicity

C. Portability

D. All of the above

Answer: D

Which of the following is NOT a Golang primitive data type?

A. int

B. float

C. bool

D. char

Answer: D

What is the keyword used to declare a new variable in Golang?

A. var

B. let

C. const

D. set

Answer: A

Which of the following is NOT a Golang control structure?

A. if-else statements

B. for loops

C. while loops

D. switch statements

Answer: C

Which of the following is a valid Golang operator?

A. &&

B. |||

C. ///

D. %%

Answer: A

What is the purpose of the fmt package in Golang?

A. To provide input/output functionality

B. To provide networking functionality

C. To provide file system access

D. To provide string manipulation functions

Answer: A

Which of the following is the correct syntax for a Golang function?

A. func name(parameters) return_type {

B. function name(parameters) return_type {

C. func name(parameters) {

D. function name(parameters) {

Answer: A

What is the purpose of a package in Golang?

A. To group related functions and types together

B. To provide an interface for interacting with the file system

C. To handle networking and communication

D. To provide advanced data structures

Answer: A

Which of the following is a standard library package in Golang?

A. net/http

B. math/vector

C. database/sql

D. io/fs

Answer: A

Which of the following is a Golang string interpolation syntax?

A. printf("Hello, %s!", name)

B. "Hello, ${name}!"

C. fmt.Sprintf("Hello, %s!", name)

D. "Hello, " + name + "!"

Answer: C

Which of the following is the correct syntax for declaring an array in Golang?

A. arr := [5]int{1, 2, 3, 4, 5}

B. arr := int[5]{1, 2, 3, 4, 5}

C. arr := []int{1, 2, 3, 4, 5}

D. arr := new [5]int{1, 2, 3, 4, 5}

Answer: A

Which of the following is the correct syntax for declaring a slice in Golang?

A. s := make([]int, 5)

B. s := [5]int{1, 2, 3, 4, 5}

C. s := []int{1, 2, 3, 4, 5}

D. s := new []int{1, 2, 3, 4, 5}

Answer: C

Which of the following is a valid way to iterate over a slice in Golang?

A. for i := 0; i < len(s); i++ {}

B. for i, val := range s {}

C. for _, val := range s {}

D. All of the above

Answer: D

Which of the following is the correct syntax for declaring a struct in Golang?

A. type Person struct {name string, age int}

B. struct Person {name string, age int}

C. type Person {name string; age int}

D. type Person struct {name string; age int}

Answer: D

Which of the following is NOT a built-in Golang interface?

A. error

B. fmt.Stringer

C. io.Reader

D. network.Conn

Answer: D

Which of the following is a correct way to implement the fmt.Stringer interface in Golang?

A. func (p Person) String() string {}

B. func (p *Person) String() string {}

C. func String(p Person) string {}

D. func String(p *Person) string {}

Answer: B

What is the correct syntax for importing a package in Golang?

A. include "package_name"

B. import "package_name"

C. use "package_name"

D. require "package_name"

Answer: B

Assignment: Write a "Guess the Number" game in Golang. The game should generate a random number between 1 and 100, and prompt the user to guess the number.

If the user's guess is too low, the program should print "Too low!" and prompt the user to guess again. If the user's guess is too high, the program should

print "Too high!" and prompt the user to guess again. When the user guesses the correct number, the program should print "You win!" and exit. The game should also keep track of the number of guesses the user has made and print it at the end of the game.

```
package main

import (
    "fmt"
    "math/rand"
    "time"
)

func main() {
    // Initialize a random seed
    rand.Seed(time.Now().UnixNano())

    // Generate a random number between 1 and 100
    secretNumber := rand.Intn(100) + 1

    // Initialize the number of guesses made by the user to zero
    numGuesses := 0

    // Prompt the user to guess the number
    var guess int
    for {
        fmt.Print("Guess the number between 1 and 100: ")
        _, err := fmt.Scanf("%d", &guess)
        if err != nil {
            fmt.Println("Invalid input. Please enter a number.")
            continue
        }

        numGuesses++
        if guess < secretNumber {
            fmt.Println("Too low!")
        } else if guess > secretNumber {
            fmt.Println("Too high!")
        } else {
            fmt.Printf("You win! You guessed the number in %d guesses.\n", numGuesses)
            break
        }
    }
}
```

This program uses the math/rand package to generate a random number between 1 and 100, and the time package to initialize a random seed for the random number generator. It then uses a for loop to repeatedly prompt the user to guess the number, and keep track of the number of guesses the user has made.

The program uses fmt.Scanf() to read the user's input from the console, and checks if the input is a valid number using error handling. If the user's guess is too low, the program prints "Too low!" and prompts the user to guess again. If the user's guess is too high, the program prints "Too high!" and prompts the user to guess again. When the user guesses the correct number, the program prints "You win!" and exits, along with the number of guesses the user made.

You can run this program in your local development environment by saving the code to a file named guess.go and running the command go run guess.go in your terminal or command prompt.