Module 4: Panics & Maps

Lesson 4.1 - Handling Panics

This lesson briefly touches upon the concept of panics in Go, which are runtime errors that can occur. While not extensively covered, it emphasizes the importance of having self-documenting return values, including errors, as a good practice in Go programming.

Key Takeaways

- Go functions can return multiple values, a common pattern being to return the result and an error value.
- Clearly documenting the return values of a function, including the meaning of each returned value and potential errors, improves code understanding and usage.

Key Concepts

- **Panic:** A built-in function that stops normal execution of the current goroutine and starts a panic sequence. Panics are usually caused by runtime errors.
- **Error Return:** The idiomatic way to signal that an operation failed in Go is by returning a value of the built-in error type as one of the function's return values. This allows the caller to handle the error explicitly.

Lesson 4.2 - Calculating Word Frequency, Working with Maps

This lesson demonstrates how to calculate the frequency of words in a text file using Go. It covers reading a file line by line using **bufio.Scanner**, using regular expressions from the **regexp** package to extract words, and utilizing maps (**map[string]int**) to store and count the word occurrences.

Key Takeaways

- The **bufio.Scanner** is useful for efficiently reading large files line by line or word by word.
- The **regexp** package provides powerful tools for pattern matching in text using regular expressions.
- Maps (map[keyType]valueType) are a key-value data structure ideal for counting the occurrences of items, where the word can be the key and the frequency the value.
- Iterating over a map using **for range** provides both the key and the value.

Key Concepts

- **Map:** A built-in associative data structure in Go that stores key-value pairs. Maps provide efficient retrieval of values based on their associated keys.
- **Regular Expression:** A sequence of characters that define a search pattern. The regexp package in Go allows you to compile and use regular expressions to find and manipulate text.
- **bufio.Scanner:** A type in the bufio package that provides a convenient interface for reading sequential data, such as lines from a file. It can also split the input based on different criteria (e.g., words).