--Progrank

WHY GO?

-Memory Management

-Security

-Slow Processing

Above issues in language like C,CPP.

-Solved by GO language and uses tools like Docker and Kubernetes.

-Multi-purpose languge

-compiled language

-static and strongly typed

-faster compilation

-simple and readable

-object document like

-pointers

-open source community

Difference Between var and :=

There are some small differences between the var var :=:

|  |  |
| --- | --- |
| **var** | **:=** |
| Can be used **inside**and **outside** of functions | Can only be used **inside** functions |
| Variable declaration and value assignment **can be done separately** | Variable declaration and value assignment **cannot be done separately** (must be done in the same line) |

Constant Rules

* Constant names follow the same naming rules as [variables](https://www.w3schools.com/go/go_variable_naming_rules.php)
* Constant names are usually written in uppercase letters (for easy identification and differentiation from variables)
* Constants can be declared both inside and outside of a function

Constant Types

There are two types of constants:

* Typed constants
* Untyped constants

# Go Output Functions

Go has three functions to output text:

* Print()
* Println()
* Printf()

The Print() Function

The Print() function prints its arguments with their default format.

## The Println() Function

The Println() function is similar to Print() with the difference that a whitespace is added between the arguments, and a newline is added at the end

The Printf() Function

The Printf() function first formats its argument based on the given formatting verb and then prints them.

Here we will use two formatting verbs:

* %v is used to print the **value** of the arguments
* %T is used to print the **type** of the arguments

## Formatting Verbs for Printf()

Go offers several formatting verbs that can be used with the Printf() function.

## General Formatting Verbs

The following verbs can be used with all data types:

|  |  |
| --- | --- |
| **Verb** | **Description** |
| %v | Prints the value in the default format |
| %#v | Prints the value in Go-syntax format |
| %T | Prints the type of the value |
| %% | Prints the % sign |

Go Data Types

Data type is an important concept in programming. Data type specifies the size and type of variable values.

Go is statically typed, meaning that once a variable type is defined, it can only store data of that type.

Go has three basic data types:

* **bool**: represents a boolean value and is either true or false
* **Numeric**: represents integer types, floating point values, and complex types
* **string**: represents a string value

Go Integer Data Types

Integer data types are used to store a whole number without decimals, like 35, -50, or 1345000.

The integer data type has two categories:

* **Signed integers** - can store both positive and negative values
* **Unsigned integers** - can only store non-negative values
* Go has five keywords/types of signed integers:

|  |  |  |
| --- | --- | --- |
| **Type** | **Size** | **Range** |
| int | Depends on platform: 32 bits in 32 bit systems and 64 bit in 64 bit systems | -2147483648 to 2147483647 in 32 bit systems and -9223372036854775808 to 9223372036854775807 in 64 bit systems |
| int8 | 8 bits/1 byte | -128 to 127 |
| int16 | 16 bits/2 byte | -32768 to 32767 |
| int32 | 32 bits/4 byte | -2147483648 to 2147483647 |
| int64 | 64 bits/8 byte | -9223372036854775808 to 9223372036854775807 |
| Type | Size | Range |
| uint | Depends on platform: 32 bits in 32 bit systems and 64 bit in 64 bit systems | 0 to 4294967295 in 32 bit systems and 0 to 18446744073709551615 in 64 bit systems |
| uint8 | 8 bits/1 byte | 0 to 255 |
| uint16 | 16 bits/2 byte | 0 to 65535 |
| uint32 | 32 bits/4 byte | 0 to 4294967295 |
| uint64 | 64 bits/8 byte | 0 to 18446744073709551615 |

The float data types are used to store positive and negative numbers with a decimal point, like 35.3, -2.34, or 3597.34987.

The float data type has two keywords:

|  |  |  |
| --- | --- | --- |
| **Type** | **Size** | **Range** |
| float32 | 32 bits | -3.4e+38 to 3.4e+38. |
| float64 | 64 bits | -1.7e+308 to +1.7e+308. |

**Tip:** The default type for float is float64. If you do not specify a type, the type will be float64.