# Quiz for "Variables, Values & Type"

Poäng totalt 35/36

This quiz will reinforce the concepts you are learning. By taking this quiz, you will become a stronger programmer.

✓ The smallest standalone element of a program that expresses some action to be carried out. * 1/1

- (●) statement ✓
- ( ) expression

✓ A combination of one or more explicit values, constants, variables, operators, and functions that the programming language interprets and computes to produce another value. * 1/1

- ( ) statement
- (●) expression ✓

✓ Which are "parentheses" or "parens" *

- (●) ( ) ✓
- ( ) { }
- ( ) [ ]

✓ Which are "curly braces" or "curlies" or "braces" * 1/1

- ( ) ( )
- (●) { } ✓
- ( ) [ ]

✓ Which are "brackets" * 1/1

- ( )
- { }
- [ ] ✓

✓ The "scope" of a variable is where you can access the variable, eg, write to it or read the value from it. * 1/1

- true ✓
- false

**Kommentarer**

https://en.wikipedia.org/wiki/Scope_(computer_science)

✓ A "primitive" data TYPE is one that is built into the language AND/OR just a basic data type which is built into the language * 1/1

- true ✓
- false

✓ In Go, an "int" is a primitive data TYPE * 1/1

- true ✓
- false

**Kommentarer**

*The "int" TYPE is built into the Go programming language. It is also a basic TYPE (not a composite TYPE).*

✓ In Go, a "string" is a primitive data TYPE * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

*The "string" TYPE is built into the Go programming language.*

✓ A "composite" data TYPE allows you to compose together values of other data TYPES * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

*In computer science, a composite data type or compound data type is any data type which can be constructed in a program using the programming language's primitive data types and other composite types. It is sometimes called a structure or aggregate data type, although the latter term may also refer to arrays, lists, etc. The act of constructing a composite type is known as composition*

✓ When a variable is declared in Go using the "var" keyword, and no VALUE is ASSIGNED to that variable, then the compiler assigns a default value to the variable. This is known as the "zero value" * 1/1

- (●) true ✓
- ( ) false

✓ Keywords are words that a reserved for use by the Go programming language; they have to be used in a certain way for a certain purpose. * 1/1

- (●) True ✓
- ( ) False

✓ Keywords are sometimes called “reserved words.” * 1/1

- (●) True ✓
- ( ) False

✓ You can’t use a keyword for anything other than its purpose. * 1/1

- (●) True ✓
- ( ) False

✓ In “2 + 2” the “+” is the OPERATOR * 1/1

- (●) True ✓
- ( ) False

✓ In “2 + 2” the “2”s are OPERANDS * 1/1

- (●) True ✓
- ( ) False

For finding documentation, what is the difference between documentation found at golang.org and godoc.org?

golang.org has std lib & src. godoc.org has std lib & third party packages.

**Kommentarer**

*Golang.org is the official website of the go programming language. Golang.org only has documentation for the standard library. Godoc.org has documentation for the standard library and third-party packages. The content of the documentation of the standard library is the same on both golang.org and godoc.org, though the content is formatted differently.*

✓ "package" is a keyword * 1/1

- (●) true ✓
- ( ) false

✓ "var" is a keyword * 1/1

- (●) true ✓
- ( ) false

✓ The entry point for all programs is in func main() which needs to be inside package main * 1/1

- (●) true ✓
- ( ) false

✓ The "short declaration operator" can be used anywhere in a program, including at both the package level and at the block level. * 1/1

- ( ) true
- (●) false ✓

✓ What are the three words used to describe good package names in the "effective go" document? * 1/1

- [ ] descriptive
- [x] short ✓
- [x] concise ✓
- [x] evocative ✓

✗ What is the name of the website where you can write (most) Go code online and have it run online? * ···/1

play.golang.org

**Rätt svar**

golang playground

**Kommentarer**

*The "golang playground" allows you to write Go code and run Go code online. You can find it here: https://play.golang.org/*

✓ A great place to ask questions is the "golang bridge forum" at https://forum.golangbridge.org/ * 1/1

- (•) true ✓
- ( ) false

**Kommentarer**

*The "golang bridge forum" at https://forum.golangbridge.org/ is a great place to ask questions.*

✓ When you see something like "fmt.Println()" this is calling the "Println()" function from the "fmt" package. * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

*When you see something like "fmt.Println()" this is calling the "Println()" function from the "fmt" package.*

✓ An "identifier" is the name assigned to a variable or a function or a constant. * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

*An "identifier" is the name assigned to a variable or a function or a constant.*

✓ To call a func, variable, or constant from a package, use the "package-dot-identifier" syntax. For example, like this, "fmt.Println()" * 1/1

- (●) True ✓
- ( ) False

## What is "idiomatic Go code"?

Go code written the community way.

**Kommentarer**

*When you write "idiomatic Go code" you are writing Go code which conforms to best practices for writing Go code.*

✓ Which character allows you to "throw away returns" or "send returns into the void"? Said another way, which character allows you to tell the compiler that you are not going to use a value returned by a function? * 1/1

- ○ #
- ○ @
- ◉ _ ✓
- ○ This is a trick question

**Kommentarer**

*The blank identifier is represented by the underscore character _. It serves as an anonymous placeholder instead of a regular (non-blank) identifier and has special meaning in declarations, as an operand, and in assignments.*
*https://golang.org/ref/spec#Blank_identifier*

✓ In Go, you cannot have a variable which you do not use. * 1/1

- (●) True ✓
- ( ) False

**Kommentarer**

*It is an error to import a package or to declare a variable without using it. Unused imports bloat the program and slow compilation, while a variable that is initialized but not used is at least a wasted computation and perhaps indicative of a larger bug. When a program is under active development, however, unused imports and variables often arise and it can be annoying to delete them just to have the compilation proceed, only to have them be needed again later. The blank identifier provides a workaround.*
*https://golang.org/doc/effective_go.html#blank_unused*

✓ When you see that a func has a parameter of this type "...interface{}" this is called a "variadic parameter" and it means that the func can take as many values of that type as you want to pass in. * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

*We will learn more about "variadic parameters" throughout the course!*
*https://golang.org/ref/spec#Passing_arguments_to_..._parameters*

✓ Every value in Go is also of type "empty interface" which is expressed like this: "interface{}" * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

*We will learn more about interfaces, and the empty interface, throughout this course!*
*https://golang.org/doc/effective_go.html#interfaces_and_types*

✓ A statement is an instruction that commands the computer to perform a specified action. Usually statements take up a line in a program. * 1/1

- (●) True ✓
- ( ) False

✓ An expression is a combination of one or more explicit values, constants, variables, operators, and functions that the programming language interprets and computes to produce another value. For example, 2+3 is an expression which evaluates to 5. * 1/1

- (●) True ✓
- ( ) False

✓ If I wanted to print to a string and then assign that value to a variable, I could use the "func Sprintf()" from the "fmt" package. * 1/1

- (●) true ✓
- ( ) false

✓ In Go, you can create your own TYPE * 1/1

- (●) true ✓
- ( ) false

✓ We don't say "casting" in Go, we say "conversion" * 1/1

- (●) true ✓
- ( ) false

✓ There is a language which we use to talk about the language. * 1/1

- (●) true ✓
- ( ) false

✓ When you create our own TYPE in Go, that TYPE will have an "underlying TYPE". * 1/1

- (●) true ✓
- ( ) false

**Kommentarer**

https://golang.org/ref/spec#Types

Det här innehållet har varken skapats eller godkänts av Google. - Användarvillkor - Ytterligare villkor

Google Formulär