# Beginner's Guide to Variables in START Programming

Welcome to the exciting world of programming! One of the fundamental concepts you'll encounter as you start your journey into the world of coding is variables. Variables are like containers that hold different types of information or data in a program. In this guide, we'll walk you through what variables are, why they're important, and how you can use them in your programs.

## What is a Variable?

Think of a variable as a labeled box where you can store information. This information can be numbers, text, true/false values, or even more complex data structures like lists of information.

## Types of Variables

In programming, variables can hold different types of data. Some common types include:

- **Integer**: Whole numbers, like 1, 10, -5, etc.
- **Float (Floating Point Number)**: Numbers with a decimal point, like 3.14, -0.5, 10.0, etc.
- **String**: A sequence of characters, enclosed in quotes, like "hello", "world", "123", etc.
- **Boolean**: Represents either true or false.
- **List**: A collection of items, which can be of different types, enclosed in square brackets and separated by commas, like [1, 2, "hello", True]. Lists can contain data of all the same type, or a mix of data types.

## Declaring a Variable

To use a variable in your program, you first need to declare it. Declaring a variable means telling the computer what type of data the variable will hold and giving it a name.

In many programming languages, you declare a variable by specifying its type followed by its name and then assigning a value to it using the `=` sign. However in start we have simplified this for more readable code, using the keyword `is` to assign a value. Some example variables in START can be seen below:

```
age is 25
piValue is 3.14
message is "Hello, World!"
isRaining is true
myNumbers is [1, 2, 3, 4, 5]
```

In the examples above:

- `age` is an integer variable holding the value 25.
- `piValue` is a float variable holding the value 3.14.
- `message` is a string variable holding the text "Hello, World!".
- `isRaining` is a boolean variable holding the value true.
- `myNumbers` is a list variable holding a sequence of numbers.

## Naming Conventions for Variables

When naming variables, it's important to follow certain conventions to make your code readable and understandable:

- Start with a letter or underscore (_), followed by letters, digits, or underscores.
- Never use special characters like $, @, #, etc.
- Use meaningful names that describe the purpose of the variable.
- Use camelCase or snake_case for multi-word variable names.

```
goodVariableName is 10
another_variable_name is "Your name here"
```

## Using the `write()` Function

To output the values of your variables, you can use START's built-in `write()` function. This function outputs the contents of a variable (or a string given to the function) to the console for you to see, for example:

```
write(content)
```

`content` is the information you want to output to the screen. This can be a variable, a string, or any other valid expression.

## Example Usage

```
name is "John"
write(name)
string is " is"
age is 20
write(string, age) nl
write("done")
```

The output of this code sample is:

```
John is 20
done
```

In this example:

- We declare a variable called `name` with the value being "John" (making this a string).
- We use the `write()` function to print the name out to the console.
- We declare 2 variables, `string` with value " is", and `age` with value 20.
- This time we use the `write()` function, with a comma separating 2 values, this will print both out with a space between them.
- We follow the final `write()` function with a `nl` character set, this will add a newline to the output (useful for if we want to show information on different lines).

- Finally we again use the write function to write “done”, we see this on the next line, due to the `nl` used previously.