

INTRODUCTION TO VISUAL AND INTERACTIVE PROGRAMMING

CT803-4-0-OIVIP

Topic: Data and Variables

Topic Learning Outcomes

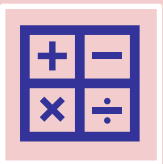
At the end of this topic, you should be able to:

- Define data types
- Define variables

Contents & Structure



Data



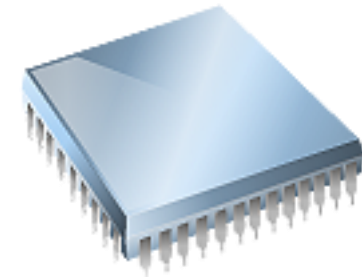
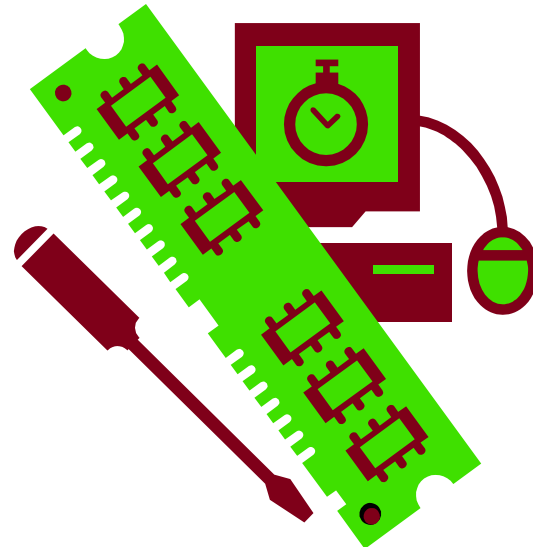
Variable

Scope of variable

Create, use, view and delete the variable

Computer Memory

A computer program needs to store all the information it is working with in the computers **RAM**. It does this using memory addresses, these addresses are numbers and because a computer is very good with numbers, this is fast and efficient.



Storing things in memory

- Humans are not good at remembering lots of number (How many phone numbers can you remember without using your mobile?).
- So to help humans, high level languages use variables to name the addresses.



Do you remember?

- What is a variable?
- How to make a new variable?
- How to change the value of the variable?
- When do you want the variable to start from 0 again?
- How to set the variable to 0?

Variables

- In programming, a variable is a placeholder for some value.
- A variable is a memory location that is used to store values .
- Variables always remember data for later use.
- A variable can be declared using a data type and a valid identifier.
- Let us first discuss data types.



What is Data Type?

- A data type defines a set of values and the operations that can be performed on that data.



Why Need Data Type?



As all of you must be aware that a computer is just a machine.



It cannot by itself distinguish between various types of Data.



This means that it cannot distinguish between the number 20 and the letter 'A' or the word "good".



For the computer all of this is just a piece of data.



It is the programmer who must tell the computer that 20 is a number, 'A' is a character and 'good' is a word.



How do we do it?

By using data types, we can classify the different data for the computer so that it can be stored and processed in a certain manner.

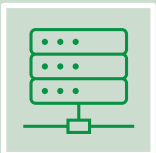
Data Types in Block Programming



Many computer programs manipulate different kinds of data, including numbers, text, images, and so on, to produce useful information.



What types of data might we need to use with our programs?



The main ones we will be using are

Numeric (numbers)

String (text)

Numeric



- Used for calculations.
- If you need to use a piece of data in a calculation you should use a numeric variable.
- Number variable can hold both integers and decimal values.



String

- These are normally letters and other nonnumeric characters.
- Sequence of characters, which can include letters (both upper- and lowercase), numbers (0 to 9), and other symbols that can type on the keyboard (+, -, &, @, and so on).
- Numbers can be stored as strings if they are not used in calculations.
 - Phone numbers
 - House numbers



Booleans

- Boolean can have only one of two values: **true** or **false**.
- Can use this data type to test one or more conditions and, based on the result, have your program choose a different execution path

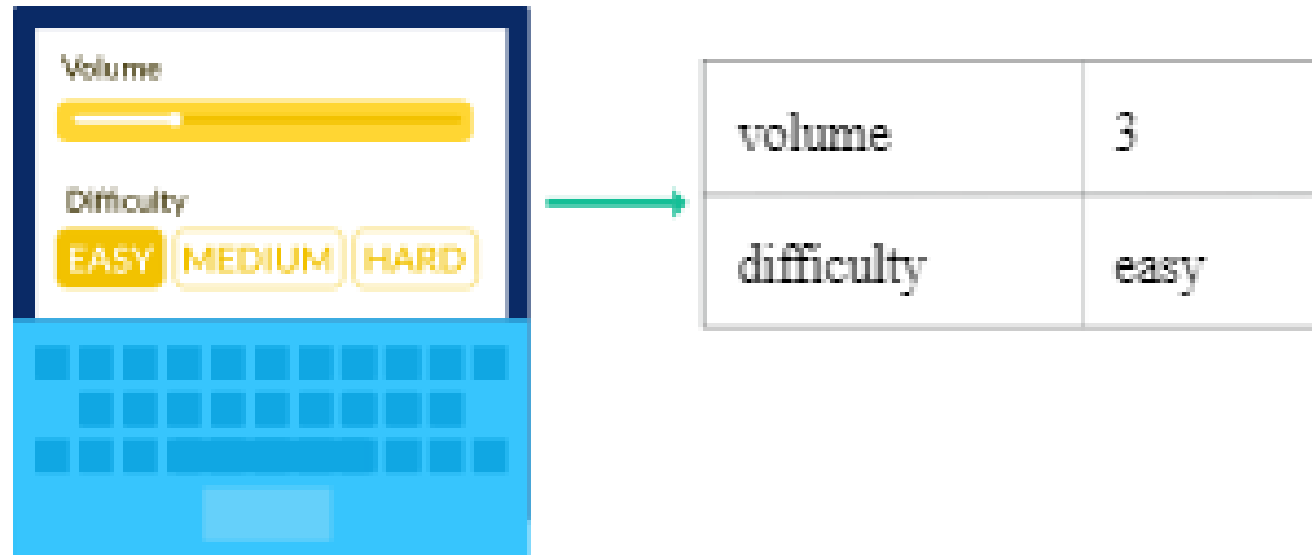
Variables in Block Programming

- In fact, in block programming, variables can contain text (strings), numbers, list, and Booleans (true/false values). Some examples are below:

name	value	type
volume	3	number
difficulty	"easy"	string
score	0	number
lives	3	number

Illustrate Your Variables

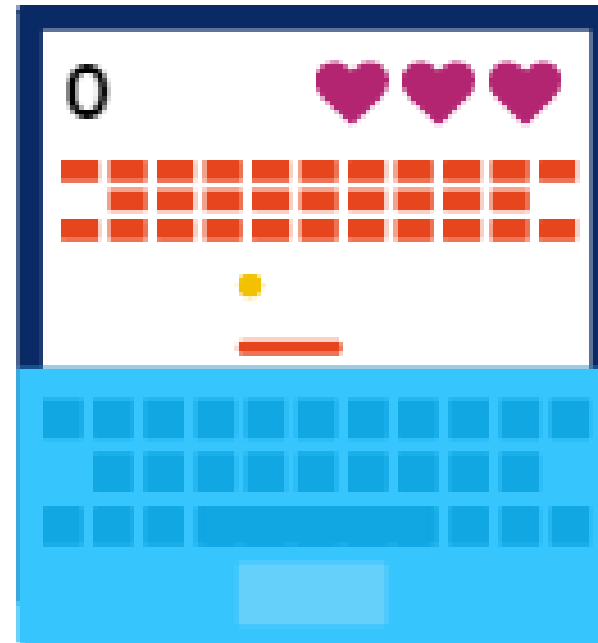
- Simple way to think about variable:
- Computer programs instruct computers how to process data. Sometimes the data comes from a user, like when a game asks you to pick a difficulty level and music volume:



Illustrate Your Variables

- Other times, the data is already stored in the program, like when that same game starts off with a score of 0 and 3 lives:

score	0
lives	3



Each variable has a name, a value, and a type. The value might change over time, and that's why its "variable."

Naming Variables

Different ways to name the variables in different programming language.

One of the popular convention is to start the name with a lowercase letter and capitalize the first letter of each additional words, such as in `sideLength`, `firstName`, and `interestRate`.

Although graphical programming allows variable names to start with numbers and contain white spaces (for example, `123Side` or `side length`), most programming languages don't. So, avoid!

Rules of Naming Variables



Highly recommend to use meaningful and descriptive name.



Single-letter variables like **w** and **z** should be kept to a minimum unless their meaning is very clear.



Avoid using long variable name.



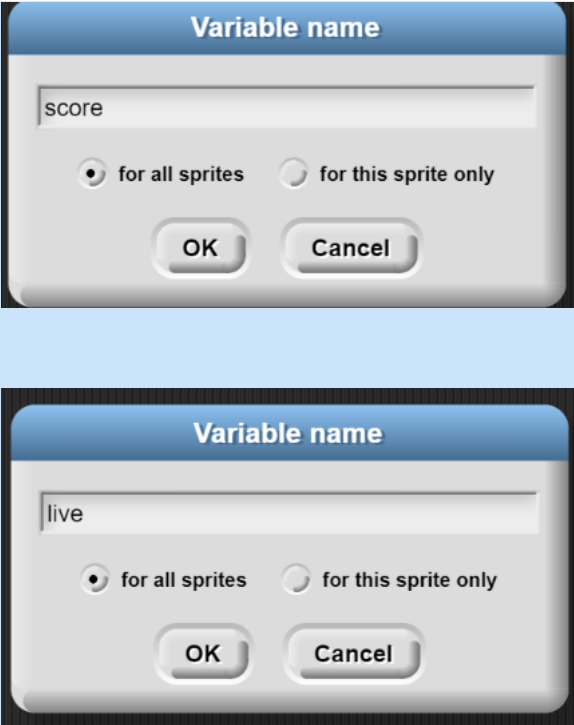
Case sensitive.

meaning that **side**, **SIDE**, and **siDE** are all unique variables



To avoid confusion, try not to use variables in the same script whose names differ only in case

Create or Declaring a variable

C++	python	graphical programming
<pre>int score = 0; int live = 3;</pre>	<pre>score = 0 live = 3</pre>	

Assigning a value to variables - Initialization

C++	python	graphical programming
<pre>int score = 0; int live = 3;</pre>	<pre>score = 0 live = 3</pre>	

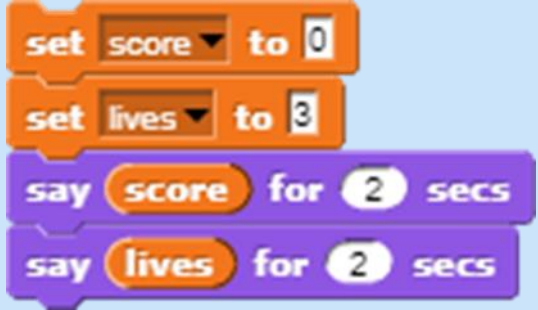
- The above code is called a statement.
- All programs are made up of statements, and each statement is an instruction to the computer what it should do.
- The code is using two statements to store two variables with two different values.
- At the back, the computer is storing each value in a different place in memory and the variables point at that memory location.

Displaying Variable

- We can get any sprite to speak by using the **say** block from inside **looks**.
- **Variables** can be displayed on the stage or **hidden**. It is often better to hide variables at the start of a program.




Displaying Variables

C++	Python	Graphical Programming
<pre>int score = 0; int live = 3; cout<< score; cout << live;</pre>	<pre>score = 0 live = 3 print(score) print(live)</pre>	 <p>The graphical programming section shows four Scratch code blocks stacked vertically. The first two are orange 'set' blocks: 'set score to 0' and 'set lives to 3'. The next two are purple 'say' blocks: 'say score for 2 secs' and 'say lives for 2 secs'.</p>

Re-assigning variables

- Variables can be re-assign to a new value using code similar to the when we assign the variable.

C++	python	graphical programming
<pre>int score = 0; int lives = 3; cout << score; cout << lives; score = 5 cout << score;</pre>	<pre>score = 0 lives = 3 print(score) print(lives) score = 5 print(score)</pre>	

Re-assigning Variables in Graphical Programming

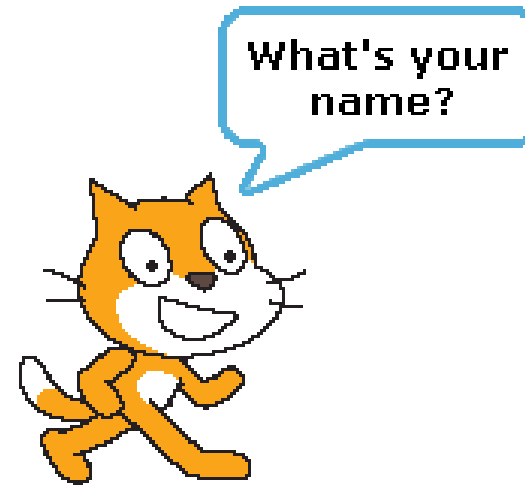
- Provides two command blocks that allow you to alter variables.
- The **set** to command directly assigns a new value to a variable, regardless of its current contents.
- The **change** by command, on the other hand, is used to change the value of a variable by a specified amount relative to its current value.

Using a String Variable

- Enter the program into **snap!**
- Remember to make the variable **name**

Answer these questions

1. What does it do?
2. How can we make it say “Hello” before Sprite asks the question and when it say’s the user’s name?



Copy this into your snap!

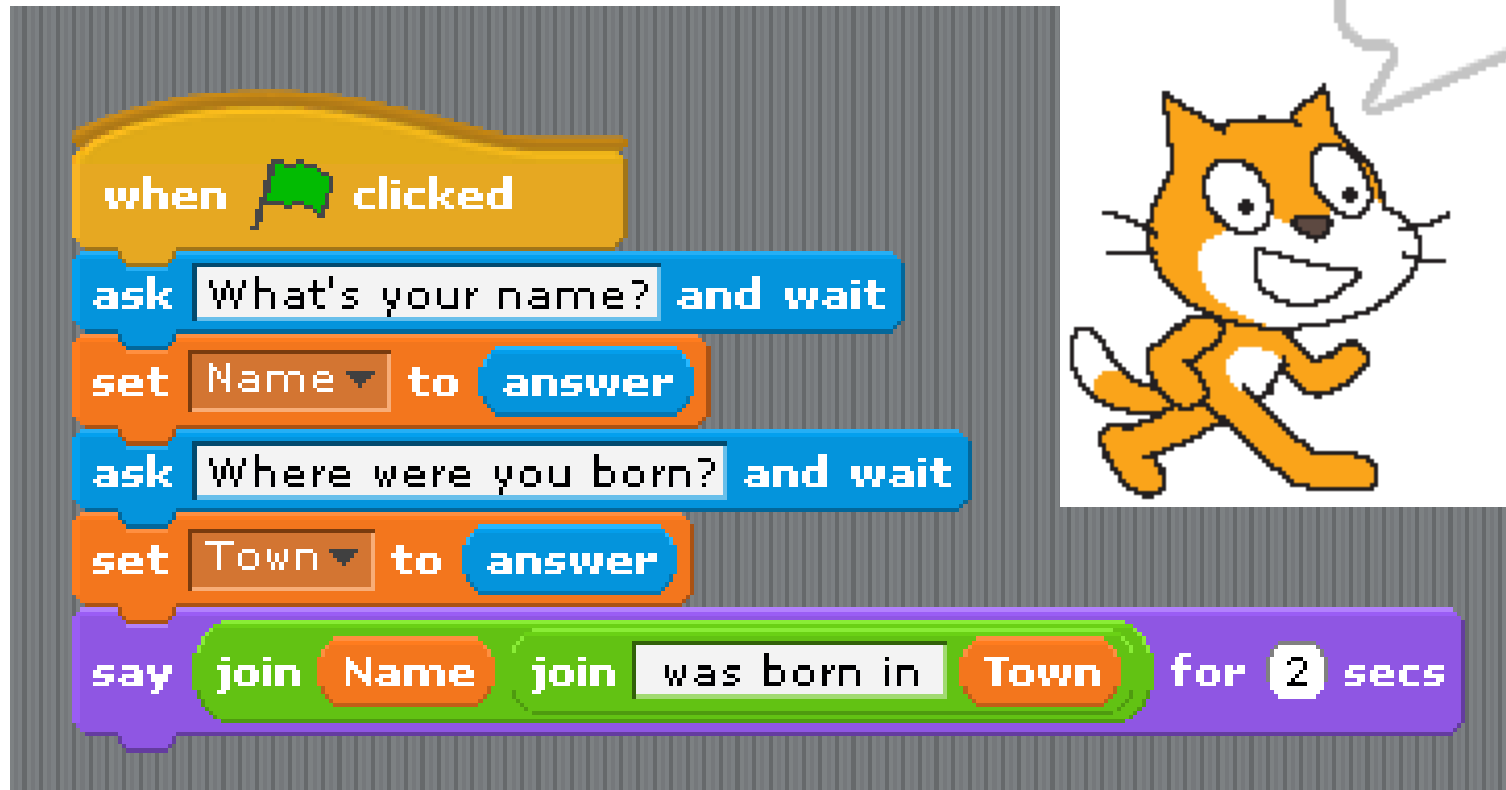


Edit your program in snap! to make it the same as above.
Write a description of what **join** does in the above program?

More than one variable

- Programs can make use of lots of variables. Just make a **new variable** in the **variable tab**
- Programmers **use sensible names** to keep track of what variable does what.
- **Answer** only stores one value at a time so remember to **set a variable to answer** after using **ask**.
- Remember the **sequence** of a program is very important, so make sure you **think** about what you want the program to do.

An example of two variables

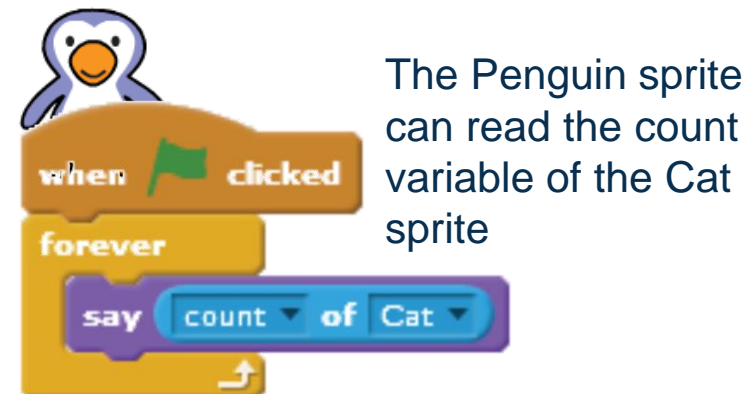
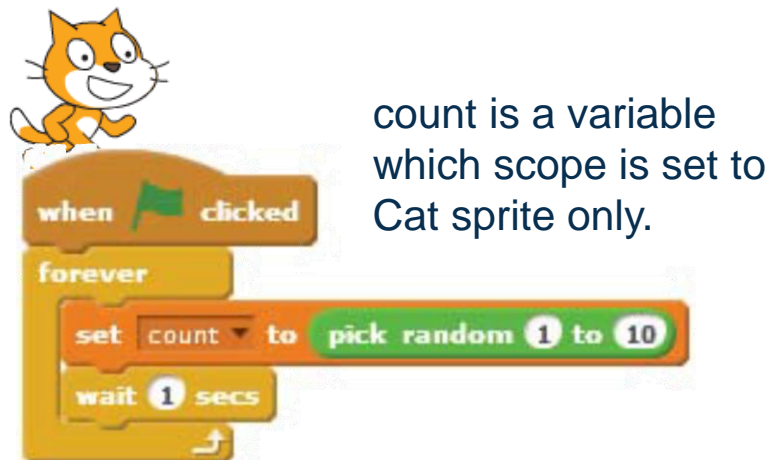


```
when clicked
ask 'What's your name?' and wait
set Name to answer
ask 'Where were you born?' and wait
set Town to answer
say join Name join ' was born in ' Town for 2 secs
```



Scope of Variables

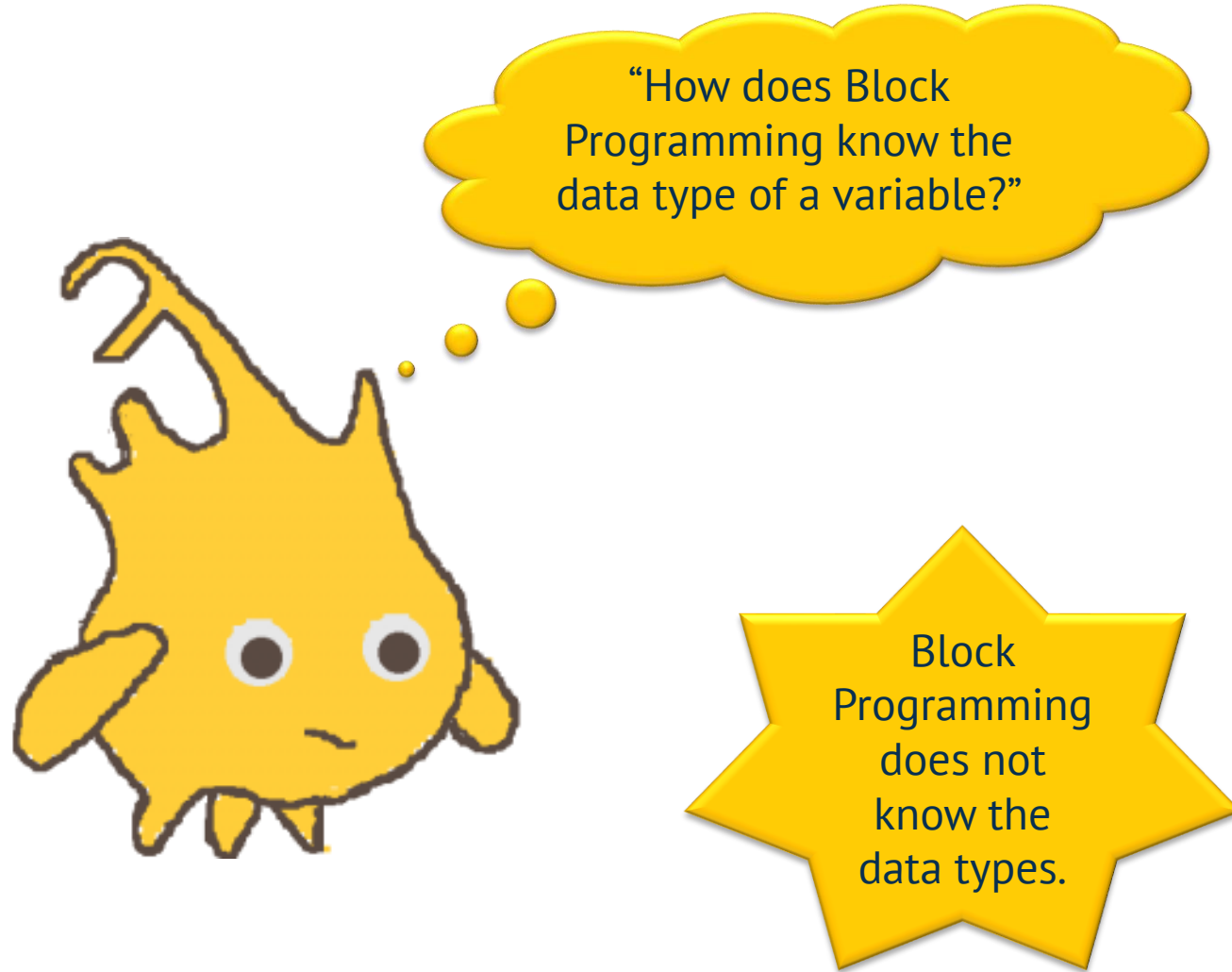
- The scope of a variable determines which sprites can write to (or change the value of) that variable.
- You can specify the scope of a variable when you create it by selecting one of the two options below:



Scope of Variables

For this sprite (Local Variable)	For all sprites (Global Variable)	Local Script Variable
<ul style="list-style-type: none"> • Scope for variables that should only be updated by a single sprite. • Different sprites can use the same name for their local variables without any conflict. (Not the value) 	<ul style="list-style-type: none"> • Can be read and changed by any sprite in your application. • For example, if a game has three buttons that allow the user to select a level to play, you can create a global variable named <code>gameLevel</code> and have each button sprite set this variable to a different number when clicked. 	<p>Scope for local variable to a single script (temporary)</p>

IMPORTANT!

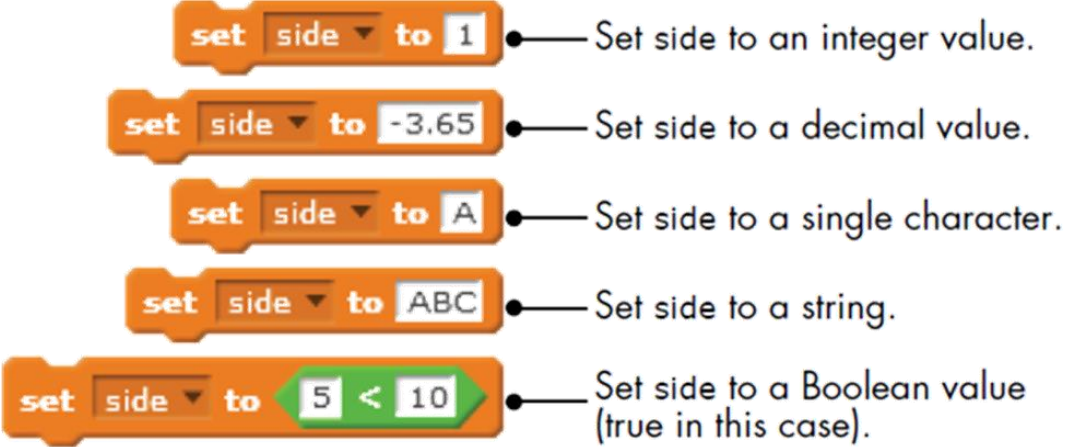


“How does Block Programming know the data type of a variable?”

Block Programming does not know the data types.

IMPORTANT!

- Any variable can hold a value of any data type.
 - For example, all of the following commands are valid in Graphical programming.
-
- Important to store the variable with correct value.



The image shows five Scratch 'set side to' blocks, each with a different value and a corresponding description:

- Block 1: 'set side to 1' — Set side to an integer value.
- Block 2: 'set side to -3.65' — Set side to a decimal value.
- Block 3: 'set side to A' — Set side to a single character.
- Block 4: 'set side to ABC' — Set side to a string.
- Block 5: 'set side to 5 < 10' — Set side to a Boolean value (true in this case).

IMPORTANT!

- Example for wrongly insert variable value.



The string "Nonsense" is converted to a number (0) and passed to the **move** command.



The string "100" is converted to a number (100) and passed to the **move** command.

- In the first script (left), Cannot convert the string "Nonsense" to a number. Rather than showing an error message, it will silently set the result of the conversion to 0 and pass this value to the move command. As a result, the sprite won't move.
- On the other hand, in the second script (right), it ignores the whitespace in the string and passes the resulting number to the move block, so the sprite moves 100 steps forward

Review Questions

- What is a variable?
- Where are variables stored?
- What command do we use in snap! to get data from a user?
- What command do we use in snap! to display information?

Tasks

- Create snap! programs to do the following.
- Display a user's name and age in a single sentence.
- Display a user's full name separately.
- Display a message from the user on the screen.

Summary / Recap of Main Points

- Data
- Variable
 - Scope of variable
 - Create, use, view and delete the variable