**Development 1: A Starting guide:**

**Chapter 1: Lesson 1:**

State: this means storage of information => Fundamental unit of information: Binding a name to a value => **CAUTION:** the label (variable name) name must be unique

And empty state, storing nothing, is a state that often occurs => a empty state is returned when a empty variable or non-existing variable is looked up

It is possible to give a variable the same state as another variable, **BUT** changing the state of the original variable will not change the state of the other variable.

A state of a variable can be altered and removed if no longer needed.

A state can contain multiple states => list, dictionary’s

Statements: a set of instructions given to a program

A combination of statements are called a program.

Statements contain 2 parts:

1. **Syntax:** structured text with constant parts => keywords => keywords are reserved words and **CANNOT** be used as a variable name, function name or any other identifier.
2. **Semantics:** describes how the statement transforms itself and a state into a new statement and a new state => gives a statement meaning

There are statements that combine two statements: **AND**

**Chapter 2: Lesson 2:**

Variable: used to bind and retrieve values in the state => variable assignment embeds the binding operation => these become bindings in the state

\*programming languages cannot be more powerful than the machine

\*\*certain states can be reached with certain language and some can’t. This makes languages more expressive than others.

**Chapter 3: Lesson 3:**

Data type: allows the developer to define data with a certain attribute that tells the compiler or interpreter how the developer intends to use this data.

A data type can be thought of as a set:

1. Contains all elements that make up the data type
2. Has important connections:
   1. These connections are structured. This structure determines a network of ties between the elements which are all and the only paths that can be followed to get to a certain point (value)
   2. These connections are also called operators
   3. When using a operator they must be given in a specific order this can be seen like following an arrow

**Expressions:** a specification to determine a value via computation => Expressions combined with statements form programs

**Evaluating expression:** following the defined structure, one piece at a time. Reduce the expression to simpler and simpler form until a value within the data type is reached

**Example:**

3 + 2 = 5 **=>** this means that we van **GO** from 3 + 2 to 5 and **NOT** 3 + 2 = 5

\*\*Computations move from a complex specification to a simpler answer but **NOT** the other wat around

**Computing:** based on following arrows