01 Zero & One

Unit 12: Programming (Python)

Programming & AI, 1st secondary (Introduction to ICT)



Engineer Rahim

12-1 Algorithm

- **Algorithms and how to draw them**: An **algorithm** is a method or set of steps for solving a particular problem.
- **Flowchart symbols**
  - **Terminal:** start / end.
  - **Display:** showing on a screen.
  - **Data:** data input and output.
  - **Process:** operations and calculations.
  - **Conditional branch:** branching by a condition (Yes / No).
  - **Repeat:** the start and end of a repetition.
- **Activity diagram symbols**
  - **Start:** and **End:** the beginning and end of the activity.
  - **Control:** a step or action that is carried out.
  - **Transition:** the arrow moving from one step to the next.
  - **Conditional branch:** branching by a condition.
  - **Parallel process:** steps that happen at the same time.
  - **Send:** sending a signal or message to the other party.
- **The three control structures**
  - **Sequential structure:** steps run in order, one after another.
  - **Repeating structure:** repeats the process while the condition is met.
  - **Branching structure:** chooses a process according to a condition
  - Any algorithm can be written with these three plus input and output.
- **Programming languages**
  - **Python:** used for AI and statistics, and runs with minimal code.
  - **JavaScript:** runs inside a web browser, so it suits anything web-related.
  - **Scratch:** a visual language for education, where you program with blocks.

Algorithm	A method or procedure for solving a particular problem.
Flowchart	A diagram that illustrates the flow of a single process.
Activity diagram	A diagram suited to representing parallel process flows.
Sequential structure	A control structure that runs processes in order.
Repeating structure	A control structure that repeats a process while a condition is met.
Branching structure	A control structure that chooses a process according to a condition.
Programming language	A language for expressing algorithms in a way a computer can understand.
Machine language	Instructions made of 0s and 1s that a computer understands directly.
Python	A language used in AI and statistics that runs with minimal code.
Scratch	A visual, block-based programming language made for education.

12-2 Programming Basics [1] (Python)

- **The print function**: `print()` displays the text or value inside the parentheses
- **Variables and the assignment operator**: A **variable** is like a box where we store data and take it back later
- **Arithmetic operators**
  - `a + b` addition, `a - b` subtraction, `a * b` multiplication.
  - `a / b` division, and its result is always a decimal number: `7 / 2` = `3.5`, and `6 / 2` = `3.0`.
  - `a // b` the whole-number **quotient**: `7 // 2` = `3`.
  - `a % b` the **remainder**: `7 % 2` = `1`.
  - `a b` **the** power **(a to the power of b):** `2  3` = `8`.
- **Sequential structure**: The program runs **line by line from top to bottom**

print()	A function that displays the text or value inside its parentheses.
Variable	A named box that stores data so it can be retrieved later.
Assignment operator	The = sign, which puts the value on the right into the variable on the left.
Arithmetic operator	A calculation symbol such as + or -.
String	A sequence of characters written between quotation marks.
Quotient (//)	The whole-number result of division.
Remainder (%)	What is left over after division.
Power (**)	a ** b means a to the power of b.

12-3 Programming Basics [2] (Python)

- **The for statement**: A **for statement** repeats a group of instructions: `for i in range(4):` with the **indented** lines under it, which are the lines that repeat.
- **range()**
  - `range(end)` : from **0** up to **end – 1**
  - `range(start, end)` : from **start** up to **end – 1**
  - `range(start, end, increment)` : goes up by increment and never reaches end
  - To count from 1 to 10 you must write `range(1, 11)`
- **Comparison operators**
  - `==` equal, and `!=` not equal.
  - `<` less than, and `>` greater than.
  - `<=` less than or equal to, and `>=` greater than or equal to.
  - Note: `=` is assignment, and `==` is comparison.
- **if and else**: A **conditional expression** returns true (True) if the condition is met and false (False) if not.
- **if ~ elif ~ else**: We use it when there is **more than one condition**

for statement	A statement that repeats the indented lines under it.
range()	Sets the values of the loop variable: start, up to end – 1, and the increment.
Indentation	Spaces before a line that show it belongs inside a for or if.
Comparison operator	An operator that compares two values (== != < > <= >=).
Conditional expression	An expression that returns True if the condition is met and False if not.

if statement	A statement that branches according to a condition.
if ~ else	Sets what happens when a condition is true and when it is false.
if ~ elif ~ else	Tests several conditions in order and runs the first true one.

12-4 Application of Programming [1] (Python)

- **Lists**: A **list** is a collection of data arranged in sequence, managed together under one name: `a = [7, 22, 11, 34, 17]` or `week = ['Sat', 'Sun', 'Mon']`.
- **Reaching the elements**: `a[0]` is the first element (7), and `a[4]` is the last element in a list of 5 elements (17)
- **Adding an element with append**: `a.append(x)` adds x to the **end** of the list
- **Two-dimensional lists**: A **two-dimensional list** is a list containing lists, with rows and columns
- **Common list programs**
  - **Total:** start with `total = 0` and add `a[i]` for each element.
  - **Minimum:** start with `m = a[0]`, and if `a[i] < m` set `m = a[i]`.
  - **Maximum:** the same idea with `>`.
  - **Counting:** `count = count + 1` every time an element meets the condition.

List	A collection of data items arranged in sequence under one name.
Element	Each value included in a list.
Index	The position number of an element, starting from 0.
append()	Adds an element to the end of a list.
Two-dimensional list	A list with rows and columns; an element is a[i][j].
a[i]	The element at index i in list a.

12-5 Application of Programming [2] (Python)

- **What is a function?**: A **function** is a group of instructions collected into one unit with a name
- **Defining and calling**: The form: `def name(arg1, arg2):` with the instructions indented under it, and `return value` at the end.
- **A function with more than one argument**: `def area(w, h):` / `return w * h` then `area(4, 3)` → 12
- **Kinds of functions**
  - **Built-in functions:** ready to use without a definition, such as `print()`.
  - **User-defined functions:** you write them yourself with `def`.

Function	A set of instructions grouped into one named unit.
Argument	A variable that receives a value inside a function.
return	Sends the function's result back to the caller.
Return value	The result a function sends back.
Function call	Running a function by its name with values for its arguments.
def	The keyword used to define a function in Python.
Built-in function	A function ready to use without a definition, such as print().
User-defined function	A function you write yourself with def.

Study the whole course free

Every lesson, questions with answers, a ministry-style bank and a timed mock exam

zerooneacademy.com

Engineer Rahim