Programming

iGCSE Computer Science

Note

For this unit, the slides only contain a teaching outline of key points for each lesson as the detailed notes for students are provided through other resources.

- Python Programming for Secondary school students (Baumgarten)
- Programming cheatsheets handout

Getting started

Setup

On your computer - Either one of the following setups:

- VS Code and Python
  - https://code.visualstudio.com/download
  - https://www.python.org/downloads/

- Thonny
  - https://thonny.org/ (ships with Python already included)

Online:

- https://repl.it/ - You will have an invite from me in your email. You will use Replit for submitting programming homework tasks.

PRIMM model

P	Predict	What will this code do?
R	Run	Test your prediction
I	Investigate	Understand & comprehend the code. Manually trace what happens. Annotate with code comments.
M	Modify	Test your understanding of how it works. What happens if?
M	Make	Create new problems using same structure and concepts

First programs

Objectives:

- Printing to screen
- Receiving input from user
- Understanding the importance of sequence
- Case sensitivity of Python
- Comments in Python
- Submitting tasks to repl.it

Numeric variables & operations

Data, types and variables

Big ideas:

- What is data?
- What is a data type?
- What are common data types?
- What is a variable?
    - Variable naming conventions

Integer variables

Objectives

- Defining an integer variable
- Printing an integer variable
- Inputting an integer variable
- Right to left assignment principle
- Basic arithmetic operations
  - Add, subtract, multiply, divide, exponent, integer divide, modulus
- Getting a random number

Repl.it practice exercises

Float variables

Objectives

- Defining a float variable
- Printing a float variable
  - Include f-string decimal formatting
- Inputting a float variable
- Basic arithmetic operations
- Convert float to integer
  - Round
  - Floor

Repl.it practice exercises

Math library

- Importing math
- math.pi
- math.sqrt()
- math.hypot()
- math.abs()
- math.sin() math.cos() math.tan() math.degrees() math.radians()

String variables & operations

Strings

Objectives

- Defining a string
- Printing a string
- Inputting a string
- Converting strings to integers or floats
- String manipulation
    - Substrings
    - len()
    - upper(), lower(), title()
    - index()
    - count()
    - replace()
    - ljust(), rjust()

repl.it exercises

Boolean variables & operations

Boolean

The boolean data-type might not initially be as intuitive to you but it is very important in computing.

Booleans represent a binary state:

- ON or OFF
- TRUE or FALSE
- YES or NO
- 1 or 0

Booleans form the basis of decision making within programs:

- If (something is true), execute this optional code
- While (something is true), continue repetitively executing this optional code loop

Booleans in Python

- Defining a boolean variable
- Printing and inputting a boolean
- Boolean comparison operations
  - < <= > >= ==
- Boolean compound operations
  - and or not

Selection

Selection

- Structure of an "if" statement
- How indentation works in Python
- if-else
- if-elif-else

Python tip - Understanding indentation
https://www.youtube.com/watch?v=1fMHH8mxo9g

Python tip: Comparing if and elif
https://youtu.be/bW9dgEFBd9I

Iteration

Iteration

- The while loop similarity to the if statement
- For loops over a simple range from 0
- Indentation reminder
- For loops with a custom start point
- For loops with a custom increment
- Totalling
- Counting

Nested statements

Nested statements

More complex programs require more than 1 selection or iteration occurring at once.

Nesting constructs within each other through use of indentation.

Exercises in repl.

Lists & arrays

Lists & arrays 1

- Why arrays?
  - Finding max, min, average of sets of values without them
- 1 dimensional arrays
- Writing into an array with iteration
- Reading from an array with iteration
- Totalling an array
- Counting an array
- Maximum, minimum, average of an array

Lists & arrays 2

- Sub arrays
- append() push() pop() remove()
- index() count()
- split() join()

Lists & arrays 3

- 2 dimensional arrays

Lists & arrays 4

Exercises

Functions

Functions

- Functions enable code reuse
- Meaningful identifier names
- Code comments
- Functions vs procedures
  - The return value
- Functions without parameters
- Functions with parameters
- Variable scope - local and global

Files & exceptions

Files

- Files enable data to be stored persistently between program executions
- Reading a text file
- Writing a text file

Exercise: Simple to-do task program

Exceptons

- Catching errors so they don't cause our program to fail