

Welcome back to CSCA08

# Computer Science

- CS is our latest attempt to address some of the deepest questions about ourselves, taking the scientific approach:
  - What is life?
  - How do we remember, and think?
  - What is learning?
  - What is a language?
  - What is consciousness?
  - ...

# Why “Computer” Science then?

- Think of computers as the labs for computer scientists; We understand things and then we recreate them in the computers, in terms of “Algorithms”
- This course is all about algorithms:
  - Algorithm: a sequence of exact and clear instructions
  - Example: IKEA instructions (for humans)

# Computer Program

- Programming languages translate algorithms to machine language (bits)
  - Compiler
  - Interpreter (Python)
- Each programming language has a set of definitions and regulations (**syntax**)
- Computer program: An algorithm written in the correct syntax
- Errors: When the computer language informs us about a mistake

# Some of the Objects Living in Python

- Pieces of basic data types:
  - Formats in which python can store and use values:
    - int, float, None, ...
- Operators
- Built-in functions
- ...

# The Abstract Model for Memory

- Each value has a certain type, and an ID:

Id1 : float



3.14

# Some Python Terminology

- *Operator, Operands:*
  - $1 + 2$        $+$  is the operator, 1 and 2 are operands
  - $-3$            $-$  is the operator, 3 is the operand
- *Python expression, evaluation:*
  - Expression is something that can be evaluated, which means can be mapped to a value that can be stored in memory and used by python.
  - When we say python evaluates something that means python ‘finds’ it, so that it could now be used
    - Evaluating a number
    - Evaluating an operator

# Evaluating Arithmetic Expressions

$$1 + -2$$

- Evaluate (find) the operator(s)
  - Find how “+” and “-” is defined and therefore how should python perform the computation / process
- Evaluate the operands
  - Find the numbers 1 and 2, so they could be used for computations
- Apply the operator to operands and evaluate the result
  - Figure out that the value of the result is -1 and Find the number -1 in memory

# Order of Arithmetic Operations (Precedence)

1. `**` : From right to left (Exponentiation)
2. `-` : Negation
3. `*`, `/`, `//`, `%` : From left to right
4. `-`, `+` : From left to right

Use parenthesis to override the order

- Examples:  
    `1 + 2 ** 3 * 4`  
    `2 ** 3 ** 2`  
    `2 ** -2`  
    `-2 ** 2`

# More Python Terminology

Function, Argument, return value:

`function_name(argument1, argument2, ...)`

- Arguments are the values provided to the function to perform the computation/process, and return the result.
- Each of the values (arguments and return value) is of a certain type

Examples:

`type(1.001)`

`id(3.14)`

`round(3.14, 1)`

`pow(5, 2)`

# Evaluating a function: function call

`max(1, -2)`

- Evaluate the function
  - Find how “max” is defined and therefore how should python perform the computation / process
- Evaluate the arguments
  - Evaluate (find) the number 1, and the arithmetic expression -2
- Perform the computation and evaluate the return value
  - Find the number 1 as the return value

# Python Documentation

- A convention for describing how objects are defined and work in python

Use special function “help” in Python Shell

```
>>> help(round)
round(number, ndigits=None)
...
```

- Documentation requires very careful reading. You will not find a single unnecessary sentence there.
- Understanding the first line is very important, it contains many details.

# Python Documentation

`round(number, ndigits=None)`

- `round` is the name of the function
- `number` refers to the first argument, with which this function is called. This argument must be present.
- `ndigits` refers to the second argument, with which this function can be called.
- The part `=None` means:

“If the second argument is provided, round will use the provided argument. Otherwise, round will use the value None for the second argument.”

This means we can call round like this: `round(42.54335)`

or like this: `round(42.54335, 2)`

i.e., The second argument is “optional”