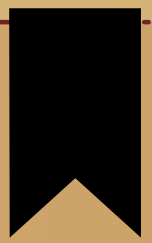
CMP 402: Machine Intelligence Course

Python
Part 1

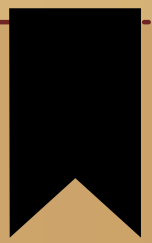
Computer Engineering Department, Faculty of Engineering
Cairo University
Fall 2016

Why python?



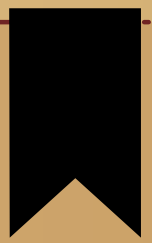
- It is easy to use (interactive with no compile-link-load-run cycle),
- Python seems to be easier to read than Lisp
- A lot free and open-source MI packages and libraries: scikit-learn, numpy, TensorFlow ....
- In Python almost every idea can be quickly validated through 20-30 lines of code

Python



- Uses indentation level rather than begin/end or braces
- Python is an interpreted, OO, HL
- Dynamic typing
- Rapid Application Development: increase productivity
- Easy to read
- Supports modules and packages
- Extensive standard library and a lot of free libraries
- Edit-test-debug cycle is incredibly fast
- GUI: Jython or Tkinter

Disadvantages



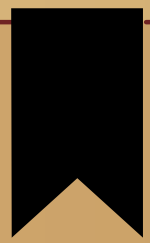
- Execution time is much slower than Lisp, often by a factor of 10
- Python is not recommended for compute intensive applications (solution : move the speed bottlenecks into C , Python/C API: https://docs.python.org/3/c-api/index.html#c-api-index
- Lisp is great for that because of it's fast prototyping and the macro utility that helps you extend the language in a great way. Because of its flexibility, Lisp has been successful as a high-level language for rapid prototyping in such areas as AI, graphics, and user interfaces.

Python 2 or 3?

- Better Unicode support (with all text strings being Unicode by default).

- Besides, several aspects of the core language (such as print and exec being statements, integers using floor division) have been adjusted to be easier for newcomers to learn and to be more consistent with the rest of the language.
- Availability of libraries
- Python 2 to 3 converters?

Language Tokens



- Comments: # , “”” multiline “””, ''' multiline'''
- Indentation (INDENT, DEDENT), how?
- Identifiers, keywords, literals, operators, and delimiters.
- Keywords:

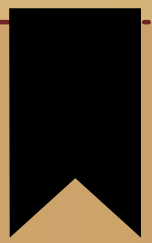
False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Language Tokens

- Literals:
  - String: 'hello', “World”
  - Integers: 7 , 2147483647, 0o177, 0b100110111, 0xdeadbeef
  - Float: 3.14 10. .001 1e100 3.14e-10 0e0
  - Imaginary: 3.14j 10.j 10j .001j 1e100j 3.14e-10j
- Operators: + - * ** / // % @ << >> & |
  ^ ~ < > <= >= == !=

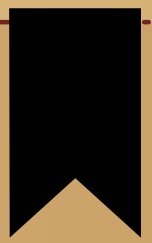
  same operator precedence like other languages (Parenthesis → Power → Multiplication → Addition → Left to Right)
- Delimiters: \ , [ ] : ...

Variables



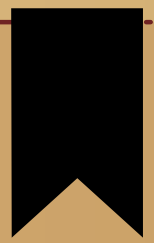
- Must start with a letter or underscore _
- May consist of letters and numbers and underscores
- Case Sensitive
- Python knows what “type” everything is
- Dynamically and strongly typed.
- Some operations are prohibited
- You cannot “add 1” to a string
- We can ask Python what type something is by using the type()

Basic Types



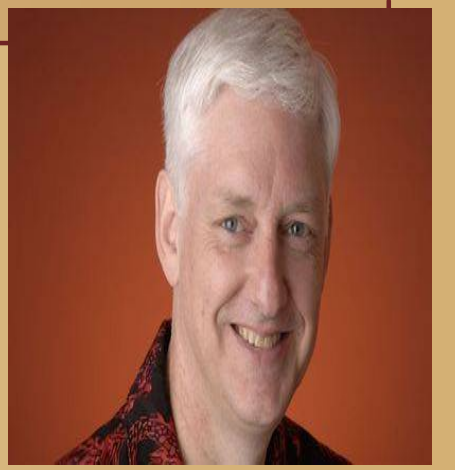
- int
- float
- bool
- string
- list

More complex types



- Two dimensional list (list of lists)
- tuple (like list but use '(' and immutable)
- dict

References



- Peter Norvig website:
  http://www.norvig.com/python-lisp.html
- Python Language Reference:
  https://docs.python.org/3/reference/index.html
- AIMA Python code (for python 3)
  https://github.com/aimacode/aima-python

Lab files

- variables_expressions.py
- agents.py
- Python Part 1 document