PYTHON

1.The paradigm of the language.

Object-oriented, imperative, functional, and procedural, and has a large and comprehensive standard library.

2. Some historical account of the evolution of the language and its antecedents.

* Python was conceived in the late 1980s, and its implementation began in December 1989by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC language (itself inspired by SETL) capable of exception handling and interfacing with the Amoeba operating system.
* Python 2.0 was released on 16 October 2000 and had many major new features, including a cycle-detecting garbage collector and support for Unicode. With this release, the development process became more transparent and community-backed.
* Python 3.0 (initially called Python 3000 or py3k) was released on 3 December 2008 after a long testing period. It is a major revision of the language that is not backward-compatible with previous versions. However, many of its major features have been backported to the backward-compatible Python 2.6.x and 2.7.x version series.
* Python 2.7's end-of-life date (a.k.a. EOL, sunset date) was initially set at 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3.
* In January 2017, Google announced work on a Python 2.7 to Go transcompiler. The Register speculated that this was in response to Python 2.7's planned end-of-life, but Google cited performance under concurrent workloads as their only motivation.
* Python 3.6 had changes regarding UTF-8 (in Windows, PEP 528 and PEP 529) and Python 3.7.0b1 (PEP 540) adds a new "UTF-8 Mode" (and overrides POSIX locale)

3. The elements of the language: reserved words, primitive data types, structured types

|  |  |  |  |
| --- | --- | --- | --- |
| **Summary of Python 3's built-in types** | | | |
| **Type** | **mutable** | **Description** | **Syntax example** |
| bool | immutable | [Boolean value](https://en.wikipedia.org/wiki/Boolean_value) | True False |
| bytearray | mutable | Sequence of [bytes](https://en.wikipedia.org/wiki/Byte) | bytearray(b'Some ASCII') bytearray(b"Some ASCII") bytearray([119, 105, 107, 105]) |
| bytes | immutable | Sequence of bytes | b'Some ASCII' b"Some ASCII" bytes([119, 105, 107, 105]) |
| complex | immutable | [Complex number](https://en.wikipedia.org/wiki/Complex_number) with real and imaginary parts | 3+2.7j |
| dict | mutable | [Associative array](https://en.wikipedia.org/wiki/Associative_array) (or dictionary) of key and value pairs; can contain mixed types (keys and values), keys must be a hashable type | {'key1': 1.0, 3: False} |
| ellipsis |  | An [ellipsis](https://en.wikipedia.org/wiki/Ellipsis_(programming_operator)) placeholder to be used as an index in [NumPy](https://en.wikipedia.org/wiki/NumPy) arrays | ... |
| float | immutable | [Floating point](https://en.wikipedia.org/wiki/Floating_point) number, system-defined precision | 3.1415927 |
| frozenset | immutable | Unordered [set](https://en.wikipedia.org/wiki/Set_(computer_science)), contains no duplicates; can contain mixed types, if hashable | frozenset([4.0, 'string', True]) |
| int | immutable | [Integer](https://en.wikipedia.org/wiki/Integer_(computer_science)) of unlimited magnitude[[75]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-pep0237-75) | 42 |
| list | mutable | [List](https://en.wikipedia.org/wiki/List_(computer_science)), can contain mixed types | [4.0, 'string', True] |
| set | mutable | Unordered [set](https://en.wikipedia.org/wiki/Set_(computer_science)), contains no duplicates; can contain mixed types, if hashable | {4.0, 'string', True} |
| str | [immutable](https://en.wikipedia.org/wiki/Immutable_object) | A [character string](https://en.wikipedia.org/wiki/Character_string): sequence of Unicode codepoints | 'Wikipedia' "Wikipedia" """Spanning multiple lines""" |
| tuple | immutable | Can contain mixed types | (4.0, 'string', True)But we can append elements using \_\_add\_\_ .  a = (4.0, 'string' , True).\_\_add\_\_(('hi' ,))  now a gives (4.0, 'string', True ,'hi') |

4. A description (in some form) of the syntax of the language.

Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use [curly brackets](https://en.wikipedia.org/wiki/Curly_bracket_programming_language) to delimit blocks, and semicolons after statements are optional. It has fewer syntactic exceptions and special cases than [C](https://en.wikipedia.org/wiki/C_(programming_language)) or [Pascal](https://en.wikipedia.org/wiki/Pascal_(programming_language)).

* Indentation- whitespace to delimit blocks
* Statements and control flow
  + The assignment statement (ex. = )
  + The if statement (executes a block code along with else and elif)
  + The for statement (iterates over object, capturing each element to a local variable for use by the attached block)
  + The try statement (allows exceptions to be caught and handled by except clauses)
  + The class statement (executes block code and attaches namespace to a class)
  + The def statement (defines function or method)
  + The with statement (encloses a code block within a context manager)
  + The pass statement (create an empty code block ; serves as NOP)
  + The assert statement (used during debugging to check for conditions that ought to apply)
  + The yield statement (returns a value from a generator function)
  + The import statement ( import modules whose functions or variables can be used in the current program) 4 ways to import
  + The print statement
* Expressions
  + +, - , \*, /
  + Matrix multiplication
  + = = (value comparison)
  + And, or, not (not symbols)
  + List comprehension
  + Lists vs tuples
  + String format operator %
  + String literals ( single or double quote marks, triple-quoted strings, raw string)
  + Array index and array slicing
* Methods
* Typing (duck typing and allows programmers to define their own types using classes) (see (3))
* Mathematics

5.The basic control abstractions of the language (loops, conditional controls, etc.)

Most Python implementations (including CPython) include a read–eval–print loop (REPL), permitting them to function as a command line interpreter for which the user enters statements sequentially and receives results immediately.

6.How the language handles abstraction (including functions, procedures, objects, modules, etc.)

7.An evaluation of the language's writability, readability, and reliability using the criteria discussed in chapter 1 and 2 of the textbook.

8. The major strengths and weaknesses of your language.

9.An overview of the programs that you included and a discussion of what language features they highlight and how the language made the programs easy/hard to implement.