Language/Syntax

- Constant
- Variable
- Rule on the name
- Reserve words
- Simultaneous/cascaded/augmented assignment
- Precedence on evaluation
- Comments: #, '''
- [ ] – assess the collections
- { } - dictionary
- Flow Control
  - Sequential
  - Selection: if, if/else, if/elif
  - Repeated: while, for
- Indentation
- try except
- break, continue
- def
- Parameter
- Argument
- Return Value
- global
- Module
  - import
  - from

Library/Method/Function

- print(), input()
- exit()
- divmod()
- float(), int(), str()
- type()
- range()
- max(), min()
- len()
- sum()
- open()
- id(), type()
- sorted()

Operator

- + / - * ** %
- =
- //
- > < ! == is
- not and or
- In
- %s, %d

- OOP
  - class
  - __init__()
  - self
  - ClassName(…)
  - . (dot operator)

Features of OOP

- Call-by-Reference
- Call-by-Value
- Encapsulation
- Inheritance
- Overriding
- Object Class
- Polymorphism
- Dynamic Binding
- Multiple Inheritance

Data Structure

- Data Type
- Boolean
- int
- float
- String
  - startswith()
  - lower()
  - upper()
  - find()
  - replace()
  - strip()
  - split()
- List
  - copy()
  - append()
  - sort()
- Dictionary
  - key, value
  - get()
  - keys(), values(), items()
- Tuple

- TextIOWrapper
  - close()
  - write()

- Data Structure
  - Compact Array
  - LinkedList
    - Singly
      - Traverse
      - addFirst
      - addLast
      - removeFirst
      - removeLast: O(n)
    - Doubly
    - Circularly
      - Round-robin scheduler

  - Linear Sequence
    - Stack
      - LIFO
      - Matching
      - LinkedList implementation
    - Queue
      - FIFO
      - List implementation
      - LinkedList implementation

  - Tree
    - Binary Tree

- Algorithm
  - Time
  - Space
  - Principles of Algorithm Analysis
  - 7 functions

- Recursion
  - Function calls itself with smaller scale
  - Recursion is implemented by Stack
  - Linear
  - Multiple

- Binary Search: Big(log n)

- Sorting Algorithms
  - Selection
  - Bubble
  - Quick

- Depth-first search
  - Recursion
  - Stack
- Breath-first search
  - Queue