Lab

Basics

List Practice

- Create a list named `a` with the elements 2, 4, 6, 8, and 10.
- Use a `for` loop to iterate over list `a` and print each item.
- Construct a `while` loop to process elements in list `a`. Utilize the `continue` statement to skip the element `6`. For elements less than 7, directly print the value. For elements greater than 7, print their square.

Ternary Operator

- Implement the following logic using a single line of code: For an integer variable `v`, assign `x` a value of 10 if `v < 3`, 20 if `3 ≤ v < 10`, and 40 if `v ≥ 10`.

Function Practice

- Write a function that accepts two parameters. The function should:
  - Concatenate the parameters if both are strings.
  - Concatenate the parameters if both are lists.
  - Add the parameters together if both are numbers.

  Test your code.

OOP Practice

- Define a class `Animal` with an attribute `name` and a method `make_sound()`.
- Create a subclass `Dog` derived from `Animal` with a modified attribute `name` and a unique method `bark()`.

Dictionary Practice

- Construct a dictionary `kvtable` with the key-value pairs: `{'a': 'x', 'b': 'y', 'c': 'z', 'd': 'w'}`.
- Create another dictionary `kvtable2` with the key-value pairs: `{'a': 'y', 'b': 'y', 'e': 'u', 'f': 'w'}`.

- Merge `kvtable2` into `kvtable` using the method `dict.update()` and analyze the resulting dictionary.

Tuple Practice

- Develop a function `foo()` designed to *compute the product of an indefinite number of inputs*.
- Create a tuple `(5, 6, 7)`, and use the function `foo()` to compute the product of the unpacked tuple values.

Magics

List Comprehension

- Use list comprehension to construct a dictionary with the structure `{1:2, 2:3, 3:4}`.

Tuple Unpacking

- Initialize `a = 1` and `b = (2, 3)`. Swap values to achieve `a = (1, 2)` and `b = 3`.

Using Lambda Expression

- Create a lambda function designed to invert the key-value pairing of a dictionary. Assume there are no duplicate values in the dictionary `{1:2, 2:3, 3:4}`.

Using F-strings

- Create a range iterator `range(4, 100, 2.2)`, iterate through it using a `for` loop, and utilize an F-string to output formatted strings like "loop_{index}, value_{value from the iterator}".

Combine Iterables

- Write a function to check if two lists `[1, 2, 3, 4]` and `[1, 2, 4, 4]` are identical using the `zip` function.
- **Advanced:** Use `.any()` and list comprehension to check list equality in a single line of code.

Context Manager

- Open and print the contents of a text file using a context manager to ensure proper handling of file operations.

Functional Programming

Map Practice

- Prepare a lambda function `lambda x: x**2` and apply it to the list `[1, 2, 3, 4, 5]` using the `map` function.

Reduce Practice

- Develop a lambda function, then use it with `reduce` to concat all strings in a list `['abc', 'bdf', '123']`, the result should be `"abc bdf 123"`, be aware of the space.
- Develop a lambda function that inverts the key-value relationship in a dictionary, suitable even for dictionaries with duplicate values in the dictionary `{1:2, 2:4, 3:4, 5:5}`.

Solutions

Original Draft