* CHAPTER-13: ADVANCED PYTHON-2

→ Virtual Environment

An Environment which is same as the system interpreter but is isolated from the other python environments on the system.

→ Installation

To use virtual Environments, we write

pip install virtualenv → install the package

we Create a new Environment using:

virtualenv myprojectenv → Create a new env

The next step after creating the virtual Environment is to activate it.

We can now use this virtual Environment as a separate python installation.

→ pip freeze Command

pip freeze returns all the packages installed in a given python environment along with the versions

"pip freeze > requirements.txt"

The above Command Create a file named requirements.t in the same directory Containing the output of pip freez

We Can distribute this file to other ~~[illegible]~~ users and they Can recreate the same environment using:

pip install -r requirements.txt

→ Lambda functions

functions Created using an Expression using lambda keyword

Syntax:

```
lambda arguments : expressions
```

↳ can be used as a normal function

Example:

```
square = lambda x : x * x
square (6)        → returns 36
sum = lambda a,b,c : a+b+c
sum (1,2,3)       → returns 6
```

→ bin method (strings)

Creates a string from iterable objects

```
l= ["apple", "mango", "banana"]

", and, ". join(l)
```

The above line will return "apple, and, orange, and, banan

→ format method (Strings)

formats the values inside the string into a desired output

```
template. format (p1, p2, ...)
```

↳ arguments

Syntax for format looks like:

```
"{} is a good {}". format ("Harry", "boy") — ①
"{1} is a good {0}". format ("Harry", "boy") — ②
```

Output for ①

Harry is a good boy.

Output for ②

boy is a good Harry

→ Map, filter & reduce

Map applies a function to all the items in an input_list

Syntax:

→ Can be Lambda function

map(function, input-list)

Filter Creates a list of items for which the function returns true.

List(filter(function))

↳ Can be a Lambda function

Reduce applies a rolling computation to sequential pair of Elements

from ~~function~~ functools import reduce

val = reduce(function, List1)

↳ Can be a Lambda functi

If the function computes sum of two numbers and the list is [1, 2, 3, 4]

```
1  2  3  4
  3   3  4     => sequential Computation
     6   4
       10
```