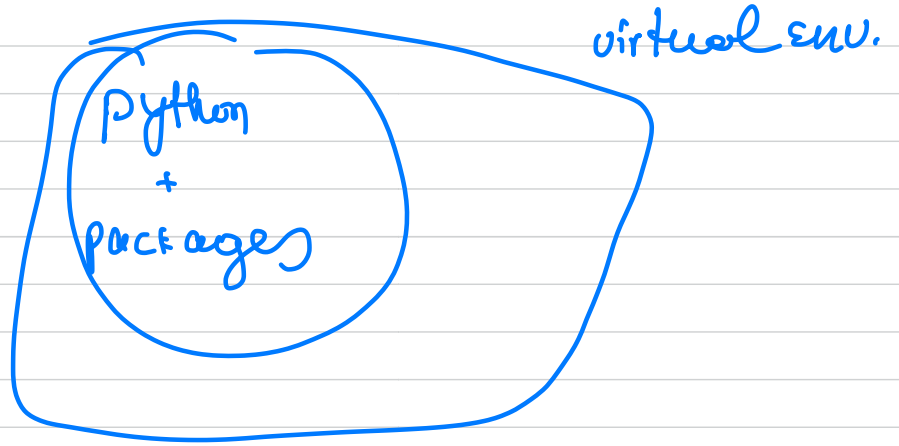


13-3-2026

Lab

Python 3.11

Virtual Environments



environment activation

Windows

CMD

```
cd Documents\dmor
```

```
dir  
: .venv
```

```
.venv\Scripts\activate
```

```
(.venv) C:\ - - -
```

```
python
```

```
import dmor
```

```
quit()
```

Παράδειγμα 1

Χωρίς modeling language

$$\max x_0 + x_1$$

$$x_0 + 2x_1 \leq 15$$

$$6 \leq 3x_0 + 2x_1 \leq 18$$

$$x_0, x_1 \geq 0.$$

Python : (highspy)

Παράδειγμα 2

Παραγωγή με αποθέματα

μεταβιβάσεις $\left\{ \begin{array}{l} x_t : \text{παραγωγή με. } t \quad t=1, \dots, T \\ I_t : \text{απόθεμα αρχή με. } t, t=2, \dots, T+1 \\ d_t : \text{ζήτηση} \quad I_1 : \text{αρχ. απόθεμα (δοσμένο)} \\ c_t : \text{κόστος παραγωγής} \\ h_t = h \quad \forall t \end{array} \right.$

$$\min \sum_{t=1}^T c_t x_t + \sum_{t=1}^T h_t I_{t+1}$$

$$(I_t = I_{t-1} + x_{t-1} - d_{t-1}, t=2, 3, \dots, T+1)$$

(ισοδ). $\rightarrow I_{t+1} = I_t + x_t - d_t, t=1, 2, \dots, T$

Θέλω παραμετρικό ορισμό των μεταβ. του Python

$$\left\{ \begin{array}{l} \text{Ορίσω } T, \quad \underline{d} = (d_1, \dots, d_T) \\ \underline{c} = (c_1, \dots, c_T) \\ h : \text{αριθμός} \end{array} \right.$$

Python class : model

Class (κατηγορία αντικειμένων)
 [

 μεθόδους (functions)

 εμπειρίες ατόμων

 πες το να αλληλ-

]

πχ. class
 integer

$x = \text{integer}() \rightarrow$

↑

μεταβλητή ενός νέου αντικειμένου (class instance)

$x = 2$

$y = \text{integer}() \quad y = 3$

2) Class student

methods [

 grades(... ..)

 mesosoros = function (grades)

 grad T/Fals =

]

Instances.

Stud1 = student()

Stud2 = student()

Stud1.grades = [6, 7, 10, ...]

Stud2.grades = [10, 8, 5, 2, ...]

$a_1 = \text{stud1, mesos_oros}$
 └── Instance └── method

$a_2 = \text{stud2.mesos_oros}$

Modeling Language

Class model

add-variable

add-constraint

set-objective

solve

display

stats

add-set

add-parameter

} δεδωτ. εξ

method

11x. Inventory.

$$T = 4$$

$$d = (5, 8, 4, 6)$$

$$c = (2, 1, 4, 2)$$

$$h = 3$$

$$I_1 = 0$$

model1 = model() ←

model1.add-parameter("d", [

" " " ("c", [

" " " ("h", 3)

model1.add-variable(x, 1, ..., T)

" " (I, ...)

def obj (model, x, I) → $\sum (c_t x_t + h_t I_{t+1})$

↳ $f(x, I) = (2x_1 + 1x_2 + \dots)$

def balance ()

model1.add_constraints(1, ..., T)