Εφαρμογές Η/Υ

1ο Μάθημα

Σχετικά με το μάθημα, εισαγωγικά για τον προγραμματισμό με Python

Διδάσκοντες

Νικόλαος Λαγαρός, Επίκουρος Καθηγητής, Εργαστήριο Στατικής & Αντισεισμικών Ερευνών

☎ 210 772 2625, ✉ `nlagaros@central.ntua.gr`

Αθανάσιος Στάμος, ΕΔΙΠ, Τομέας Δομοστατικής

☎ 210 772 3665, ✉ `stamthan@central.ntua.gr`

Χριστόδουλος Φραγκουδάκης, ΕΔΙΠ, Κέντρο Ηλεκτρονικών Υπολογιστών

☎ 210 772 2434, ✉ `chfrag@central.ntua.gr`

Σχετικά με το μάθημα:

Ποιοί είναι οι στόχοι;

- Χρήση του Ηλεκτρονικού Υπολογιστή για την επίλυση προβλημάτων Μηχανικού
- Γλώσσα Προγραμματισμού: Python
- Γενικά παραδείγματα. Προγράμματα με ειδικές μορφές εκτύπωσης. Προβλήματα γεωμετρικά και αλγεβρικά
- Αριθμητική ολοκλήρωση, ελάχιστα τετράγωνα, σειρές Fourier
- Πίνακες: αντιστροφή, ιδιοτιμές , επίλυση γραμμικών συστημάτων
- Εφαρμογές από τους τομείς:
  - Δομοστατικής
  - Υδατικών Πόρων και Περιβάλλοντος
  - Μεταφορών και Συγκοινωνιακής Υποδομής
  - Γεωτεχνικής
  - Προγραμματισμού & Διαχείρισης Τεχνικών Έργων
- Γραφικές απεικονίσεις

Σχετικά με το μάθημα:

Ασκήσεις

Στα πλαίσια του μαθήματος θα δοθούν ασκήσεις οι οποίες θα παραδίδονται σε συγκεκριμένη ημερομηνία (Προαιρετική Παράδοση)

Εργασία Εξαμήνου

Ομάδες 4 ή 5 ατόμων θα αναλάβουν ένα θέμα μεταξύ των προτεινόμενων από τους Τομείς ή δικής τους επιλογής (Υποχρεωτική Παράδοση). Ποσοστό συμμετοχής στον τελικό βαθμό 20%.

Αξιολόγηση

Οι ασκήσεις είναι προαιρετικές και θα προσμετρηθούν (μόνο θετικά) εφόσον ο βαθμός της τελικής εξέτασης είναι μεγαλύτερος ή ίσος με 4

Τρόπος Εξέτασης

Η εξέταση γίνεται σε Η/Υ. Ποσοστό συμμετοχής της τελικής εξέτασης στον τελικό βαθμό 80%

Guido van Rossum

Δημιούργησε την Python το 1991

Επηρεασμένος από τη γλώσσα ABC άρχισε να προγραμματίζει την Python σαν hobby το 1989

Ήταν fan της κωμικής σειράς Monty Python's Flying Circus

ACM Distinguished Engineer (2006)

Το 2012 μετακινήθηκε από την Google στην Dropbox

Στόχοι του van Rossum για την Python

- Εύκολη, να ακολουθεί τη διαίσθηση και εξίσου δυνατή με τον "ανταγωνισμό"
- Ανοικτού κώδικα, ώστε ο καθένας να συνεισφέρει στην ανάπτυξή της
- Κώδικας κατανοητός, σχεδόν σαν καθομιλουμένη
- Για καθημερινή χρήση, άμεση ανάπτυξη της ιδέας σε πρόγραμμα

Η Python είναι ιδιαίτερα δημοφιλής

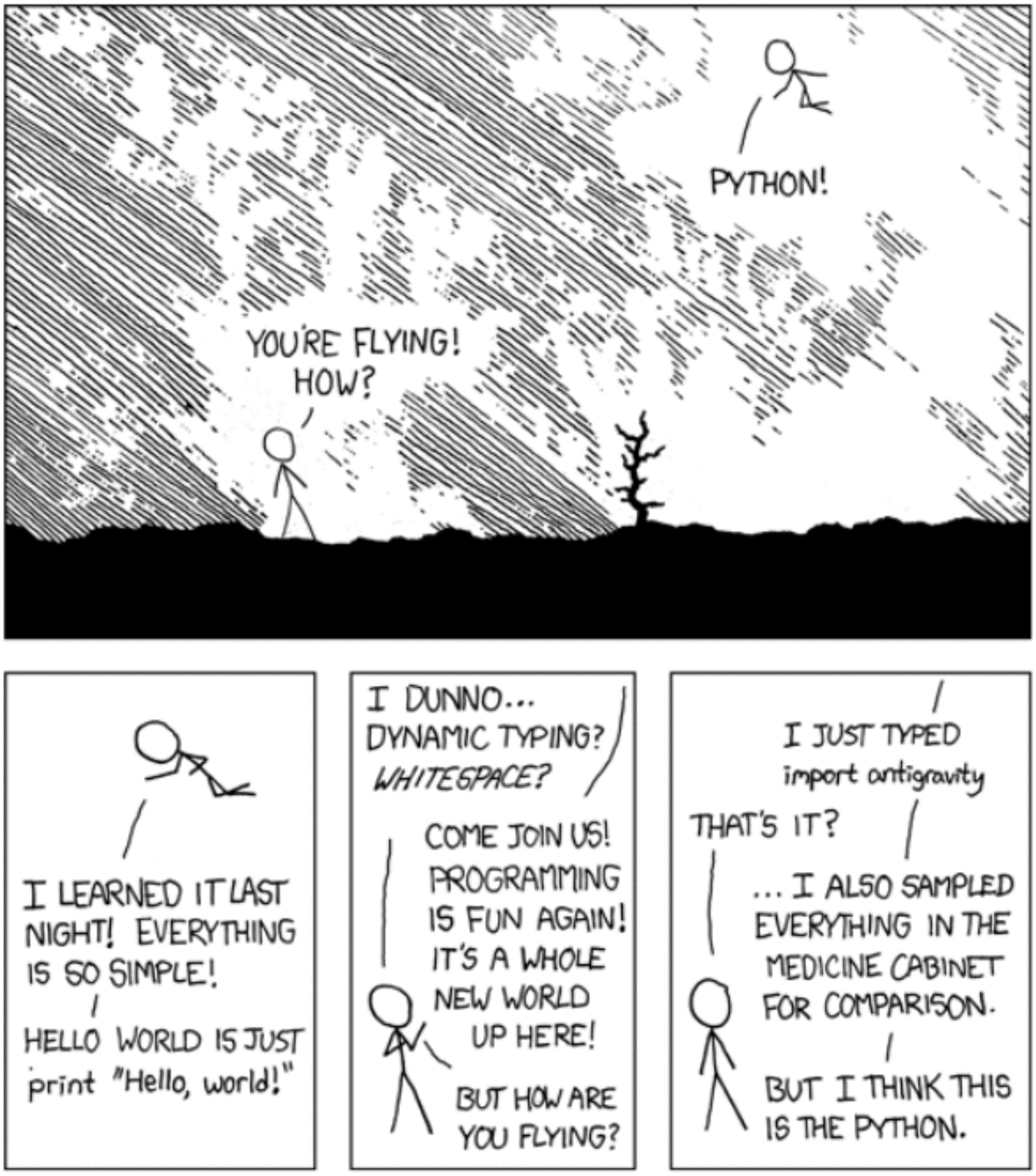
- Η τρίτη δημοφιλέστερη γλώσσα στο Github
- Μέσα στις 10 πιο δημοφιλείς γλώσσες στις αγγελίες στο εξωτερικό
- Η πιο δημοφιλής γλώσσα στα πανεπιστήμια της Αμερικής
- Χρήστες μεταξύ άλλων:
  - Google
  - Dropbox
  - Spotify, Netflix
  - NASA
  - YouTube
  - ArcGIS
  - SIMULIA Abaqus
- Περιέχεται εξ ορισμού σε κάθε διανομή Linux και στο Mac OS
- Υποστηρίζεται πολύ καλά στα Windows

The Zen of Python

Η Python έχει μια ενθουσιώδη κοινότητα που έχει συνολικά διαμορφώσει μια συγκεκριμένη άποψη για το προγραμματιστικό στύλ. Οι βασικές αρχές είναι γραμμένες μέσα στην ίδια τη γλώσσα:

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
...
Readability counts.
...
There should be one-- and preferably only one --obvious way to do it.
...
Now is better than never.
...
>>> import antigravity
```


PYTHON!
YOU'RE FLYING!
HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Τι είναι η Python;

- Γλώσσα προγραμματισμού υψηλού επιπέδου (high-level language for general and technical computing).
- Υπολογισμοί, απεικόνιση, προγραμματισμός σε φιλικό περιβάλλον (an easy-to-use environment).

Τυπική χρήση

- Μαθηματικά και υπολογισμοί και Ανάπτυξη αλγορίθμων
- Προσομοίωση
- Ανάλυση δεδομένων, απεικόνιση, Γραφήματα
- Εφαρμογές διαδικτύου (web applications)
- Παίγνια
- Ανάπτυξη εφαρμογών, συμπεριλαμβανομένου περιβάλλοντος χρήσης (Graphical User Interface-GUI)

Γιατί Python;

Αποτελεί μια καλή επιλογή για την ανάπτυξη προγραμμάτων. Όπως και το Matlab:

- Εύκολη και πολύ ταχεία προτυποποίηση.
- Γρήγορη εκμάθηση και καλή τεκμηρίωση.
- Διαθέτει καλές βιβλιοθήκες συναρτήσεων και επεξεργασίας εικόνας.
- Εξαιρετικές δυνατότητες απεικόνισης.
- Χρησιμοποιείται ευρέως για τη διδασκαλία και την έρευνα στα πανεπιστήμια και τη βιομηχανία!
- Πχ το Dropbox είναι γραμμένο σε Python

Γιατί Python;

Επιπλέον διαθέτει πλούσιους τύπους αριθμητικών δεδομένων

- Ακέραιους, πραγματικούς, μιγαδικούς αριθμούς (και μητρώα/διανύσματα από αυτούς)
- "Δεκαδικούς" αριθμούς (για οικονομικά)
- Κλασματικούς αριθμούς, πραγματικούς με Ν δεκαδικά, και αυθαίρετα μεγάλους ακεραίους (long)
- Αυτόματη μετατροπή ακεραίων σε long, ή πραγματικούς (ξεχωριστή σύνταξη για ακέραια διαίρεση)

Γιατί Python;

Το πρόγραμμα Hello World! σε τρεις γλώσσες προγραμματισμού:

- C++:

```
#include <iostream.h>
void main() {
    cout << "Hello, world!" << endl;
}
```

- Java:

```
public class HelloWorld {
    public static void main(String[] args) {
            System.out.println("Hello World!");
    }
}
```

- Python:

```
print("Hello world!")
```

Γιατί Python;

Μοιάζει με ψευδοκώδικα

Παραγοντικό

```
def factorial(n):
    return 1 if n==1 else n*factorial(n-1)
```

Διαφορά συνόλων

```
def set_difference(set1, set2):
    return [x for x in set1 if x not in set2]
```

Εκτύπωση του αρχείου `file` στην οθόνη

```
for line in open('file'):
    print(line)
```

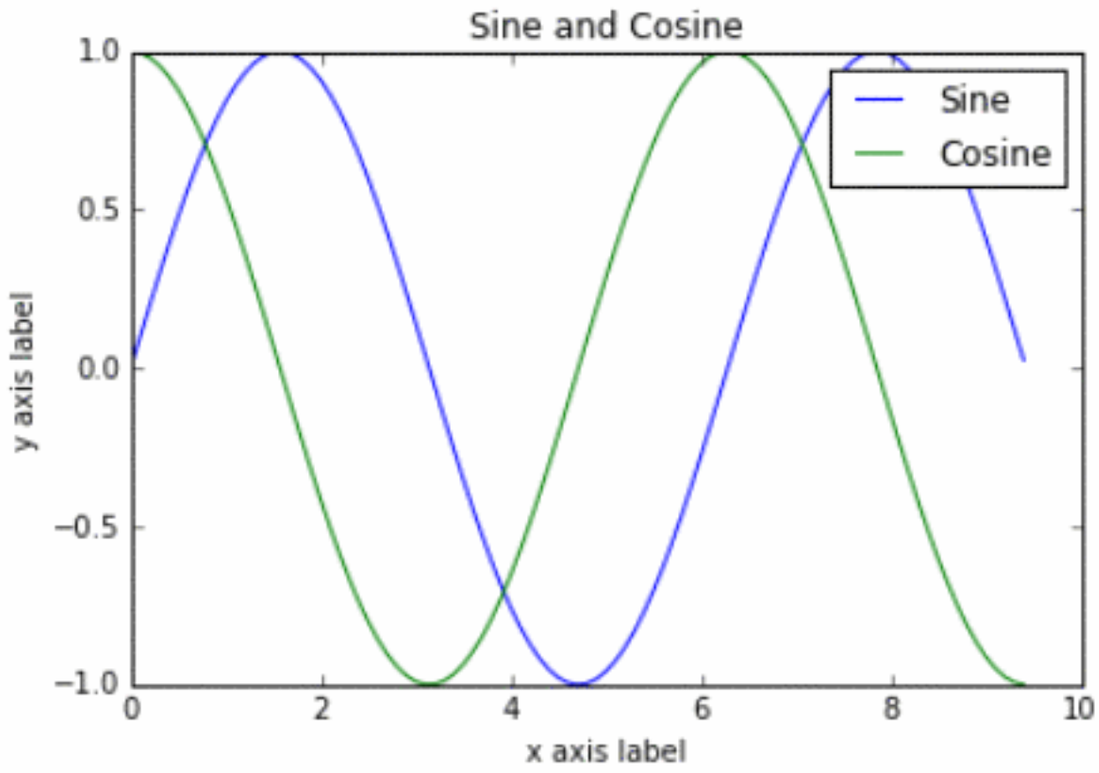
Γιατί Python;

Υψηλής ποιότητας βιβλιοθήκες για γραφικές παραστάσεις

```
import numpy as np
import matplotlib.pyplot as plt

# Compute the x and y coordinates
x = np.arange(0, 3 * np.pi, 0.1)
y_sin = np.sin(x)
y_cos = np.cos(x)

# Plot the points using matplotlib
plt.plot(x, y_sin)
plt.plot(x, y_cos)
plt.xlabel('x axis label')
plt.ylabel('y axis label')
plt.title('Sine and Cosine')
plt.legend(['Sine', 'Cosine'])
plt.show()
```



Γιατί Python;

Διαθέσιμες βιβλιοθήκες σχεδόν για οτιδήποτε

Ένα πρόγραμμα Python που στέλνει email:

```
import smtplib
from email.mime.text import MIMEText

server = smtplib.SMTP('smtp.ntua.gr')
server.starttls()
server.login("YOUR_USER_NAME", "YOUR_SUPER_SAFE_PASSWORD")

msg = MIMEText("Hello from Python!")
msg['Subject'] = "You have a message from Python!"
msg['From'] = "My Python script"
server.sendmail("chfrag@central.ntua.gr", "chfrag@gmail.com", msg.as_string())
server.quit()
```

Γιατί Python;

Διαθέσιμες βιβλιοθήκες σχεδόν για οτιδήποτε

Ένα πρόγραμμα Python που υλοποιεί έναν HTTP server:

```
from http.server import BaseHTTPRequestHandler, HTTPServer

class testHTTPServer_RequestHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.send_header('Content-type','text/html')
        self.end_headers()
        message = "Hello world!"
        self.wfile.write(bytes(message, "utf8"))
        return

server_address = ('127.0.0.1', 8081)
httpd = HTTPServer(server_address, testHTTPServer_RequestHandler)
httpd.serve_forever()
```

Γιατί Python;

PyPI - the Python Package Index

`https://pypi.python.org/pypi`

Είναι ένα online αποθετήριο "πακέτων Python" που σήμερα περιέχει 89.863 πακέτα. Η χρήση των πακέτων είναι άμεση και απλή με το εργαλείο `pip`:

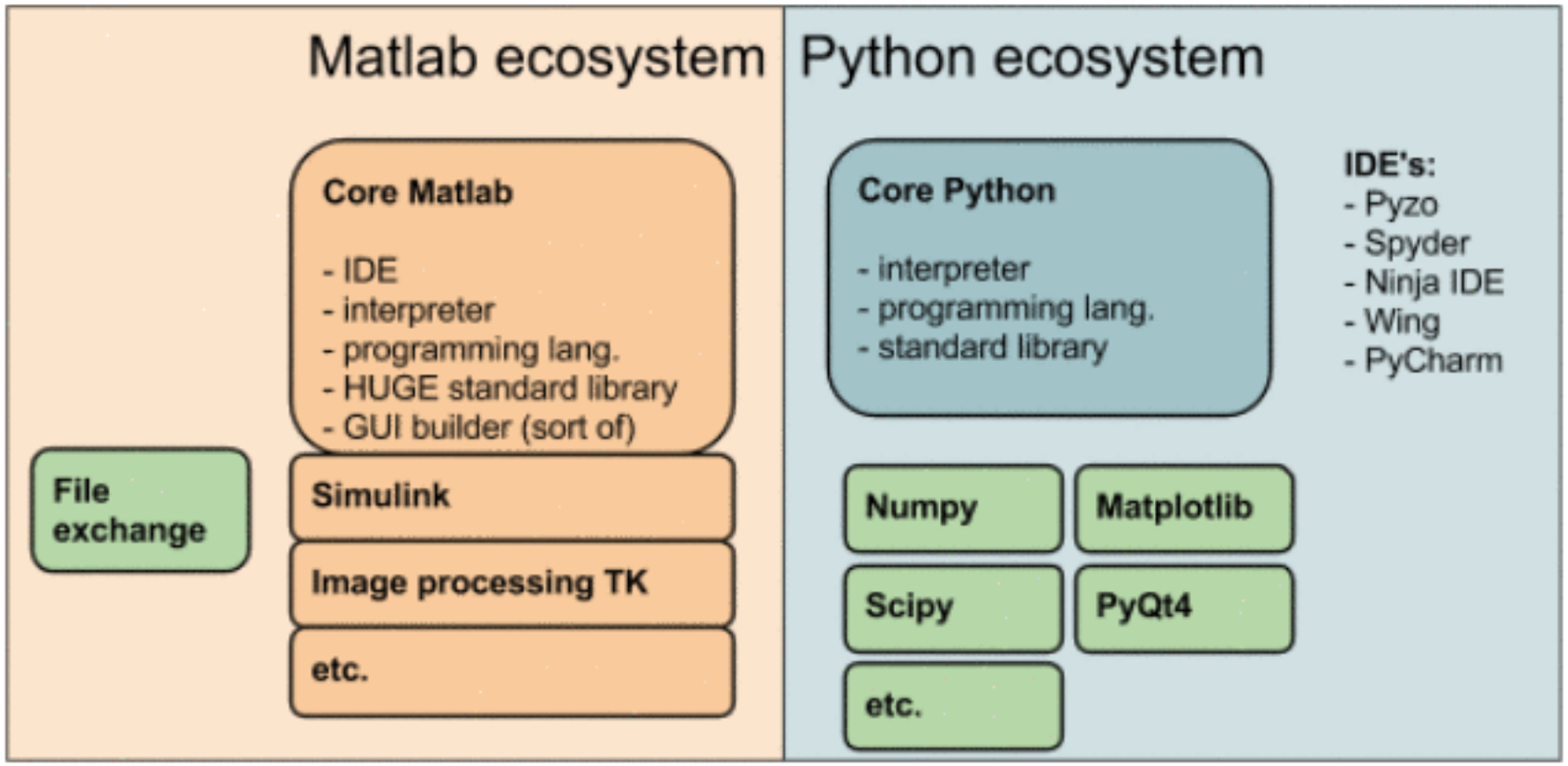
```
pip install package
```

Το πακέτο package είναι άμεσα διαθέσιμο πρός χρήση:

```
import package
# use the package
...
```

Γιατί Python;

Τόσο δυνατή όσο το Matlab (περίπου)



- Δεν υπάρχει κάτι αντίστοιχο του Simulink

Γιατί Python;

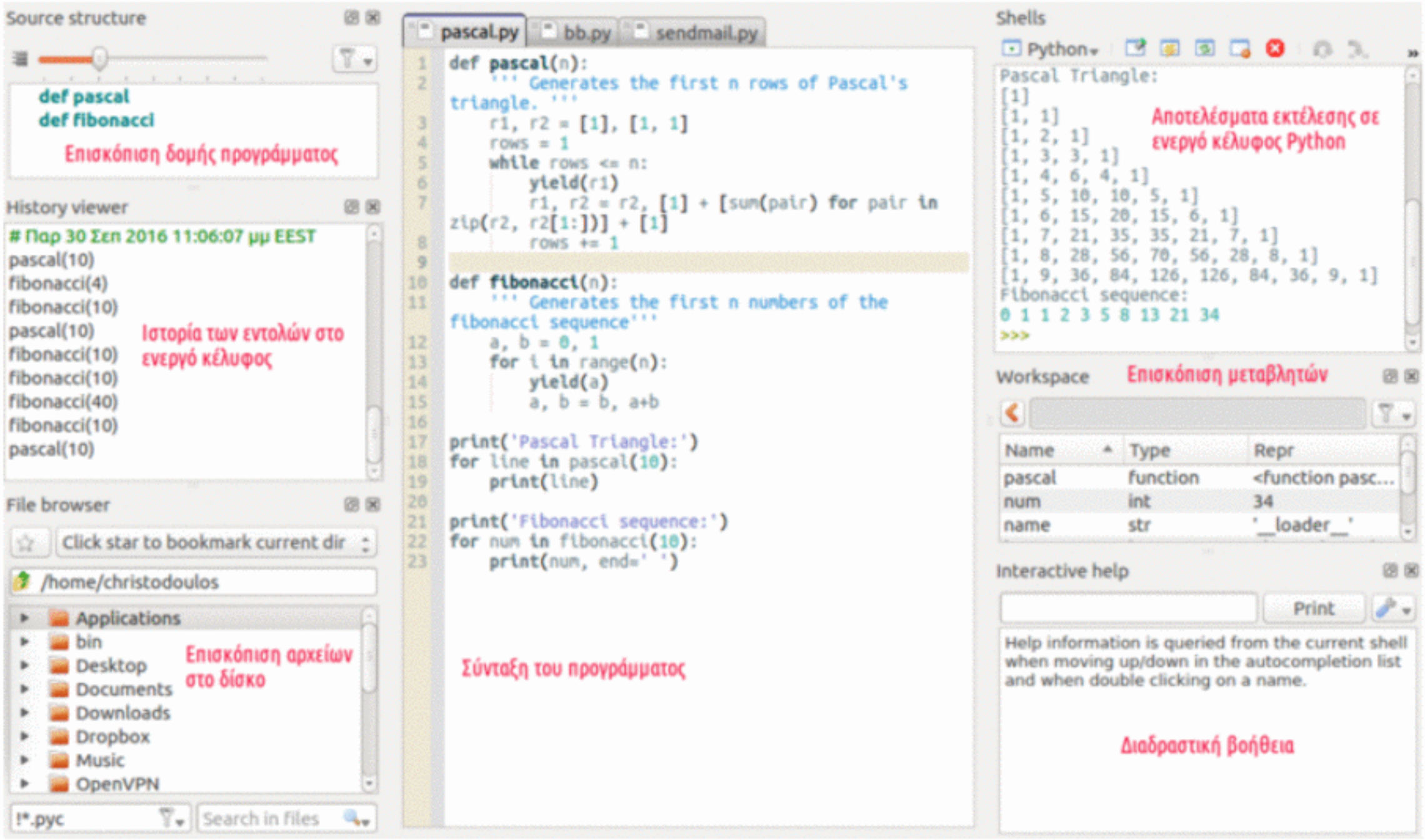
Πλεονεκτήματα έναντι του Matlab

- Είναι ελεύθερο λογισμικό
- Όμορφη προγραμματιστική γλώσσα, κομψός κώδικας
- Εξίσου δυνατή, πληθώρα βιβλιοθηκών για το οτιδήποτε
- Δυνατές δομές δεδομένων (λίστες, σύνολα, λεξικά)
- Namespaces, Introspection
- Δυνατός χειρισμός συμβολοσειρών

```
"I code in Matlab".replace('Matlab', 'Python').rjust(30)
```

- Άμεση φορητότητα μεταξύ Windows, Linux, OS X
- Εγγενώς αντικειμενοστραφής
- Δυνατές εργαλειοθήκες για προγραμματισμό GUI (Wx, Qt, GTK)

Το ολοκληρωμένο περιβάλλον Pyzo



Εισαγωγή στην Python

Χρήση σαν αριθμομηχανή

```
>>> 2 + 2 # Αυτό είναι ένα σχόλιο
4
>>> (50 - 5 * 6) / 4
5.0
>>> 8 / 5
1.6
>>> 7 // 3 # Ακέραια διαίρεση
2
>>> 7 % 3  # Υπόλοιπο διαίρεσης
1
>>> 7 // -3
-3
>>> 7 % -3
-2
```

- Χρήση των `+`, `-`, `*`, `/`, `%` κατά τα γνωστά
- Ακέραια διαίρεση: $\frac{a}{b} = q + r$ έτσι ώστε $b \cdot q + r = a$ και $0 \leq r < b$

Εισαγωγή στην Python

Μεταβλητές

```
>>> width = 20
>>> height = 5 * 9
>>> width * height
900
>>> x = y = z = 0      # Ανάθεση μιας τιμής σε πολλές μεταβλητές
>>> x, y, z = 1, 2, 3  # Ανάθεση πολλών τιμών σε πολλές μεταβλητές
>>> x
1
>>> y
2
>>> z
3
>>> y, z = z, y        # Οι τιμές των y και z ανταλλάσσονται
>>> y
3
>>> z
2
```

Εισαγωγή στην Python

Ανάθεση στις μεταβλητές

```
>>> x = 3     # Ένα όνομα εμφανίζεται πάντα στα αριστερά της ανάθεσης
>>> id(3)     # Η συνάρτηση id(x) επιστρέφει τη θέση μνήμης του x
10911776
>>> id(x)
10911776
>>> y = x
>>> id(y)
10911776
>>> n
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'n' is not defined
```

- Η ανάθεση ορίζει ότι ένα όνομα αναφέρεται σε ένα αντικείμενο.
- Η ανάθεση δημιουργεί αναφορά και όχι αντίγραφο.
- Τα ονόματα στην Python δεν έχουν τύπους. Τα αντικείμενα έχουν τύπους.
- Ο τύπος της αναφοράς συνάγεται αυτόματα από τον τύπο του αντικειμένου.
- Ένα όνομα δημιουργείται την πρώτη φορά που εμφανίζεται σε μια ανάθεση.
- Ένα όνομα πρέπει να οριστεί πριν χρησιμοποιηθεί.

Εισαγωγή στην Python

Βασικοί τύποι αντικειμένων

Ακέραιοι (integers)

```
z = 7
```

Κινητής υποδιαστολής (floats)

```
z = 5 / 2
print(z) # Τυπώνει 2.5
```

Αλφαριθμητικά (strings)

```
s1 = "John's toy"
s2 = 'Your answer was "Marry Poppins"'
s3 = """a'b"c"""
```

- Δηλώνονται ανάμεσα σε `" "` ή `' '` ή `''' '''`
- Μπορούν να περιέχουν `"` ή/και `'`

Εισαγωγή στην Python

Προτασιακός λογισμός

a	not a	a	b	a and b	a or b
True	False	True	True	True	True
False	True	True	False	False	True
		False	True	False	True
		False	False	False	False

Δύο τιμές αλήθειας: True, False

Κανόνες De Morgan:

- not (a and b) έχει την ίδια τιμή αλήθειας με το (not a) or (not b)
- not (a or b) έχει την ίδια τιμή αλήθειας με το (not a) and (not b)

Εισαγωγή στην Python

Βασικοί τύποι αντικειμένων

Λογικές εκφράσεις (Booleans)

```
>>> w, x, y, z = 1, 2, 3, 4
>>> x == 1
False
>>> w < y and x < y and y < z
True
>>> w < x < y < z                # Ισοδύναμο με το παραπάνω
>>> x < y or z == 1
True
>>> x > y or x == 4
False
>>> # Τιμές αλήθειας των εκφράσεων a and b και a or b
... True and True, True and False, False and True, False and False
(True, False, False, False)
>>> True or True, True or False, False or True, False or False
(True, True, True, False)
```

Στην αποτίμηση των λογικών εκφράσεων οι πράξεις `and` και `or` εκτελούνται τελευταίες

Εισαγωγή στην Python

Βασικοί τύποι αντικειμένων

Οκνηρή αποτίμηση λογικών εκφράσεων (lazy evaluation)

- Στο `a and b` αν `a == False` η Python δεν εξετάζει την τιμή του `b`
- Στο `a or b` αν `a == True` η Python δεν εξετάζει την τιμή του `b`
- Τα διάφορα "κενά" αντικείμενα (π.χ. `0`, `''`, `[]`) αντιστοιχούν στο `False`
- Τα "μη κενά" αντικείμενα αντιστοιχούν στο `True`
- Αν το `a` είναι "κενό αντικείμενο" το `a and b` έχει την τιμή του `a`
- Αν το `a` είναι "κενό αντικείμενο" το `a or b` έχει την τιμή του `b`

```
>>> 0 == 1 and k < l  # k και l είναι ονόματα που δεν έχουν οριστεί
False
>>> 1 == 1 or k < l   # k και l είναι ονόματα που δεν έχουν οριστεί
True
>>> 0 and 1, '' and 'a'
(0, '')
>>> 0 or 1, '' or 'a'
(1, 'a')
```

Εισαγωγή στην Python

Οι βασικοί τύποι αντικειμένων δεν μεταλλάσσονται με την ανάθεση

```
>>> x = 3       # Δημιουργείται το 3, το όνομα x αναφέρεται στο 3
>>> id(x)
10911776
>>> y = x       # Δημιουργείται το όνομα y που αναφέρεται στο 3
>>> id(y)
10911776
>>> y = 4       # Δημιουργείται το 4, αλλάζει η αναφορά του y
>>> id(y)
10911808
>>> print(x)    # Το x δεν έχει επηρεαστεί, συνεχίζει να αναφέρεται στο 3
3
```

- Οι τύποι που δεν μεταλλάσσονται με την ανάθεση λέγονται immutable
- Οι τύποι που μεταλλάσσονται με την ανάθεση λέγονται mutable (θα τους δούμε στη συνέχεια: λίστες, λεξικά, τύποι ορισμένοι από το χρήστη)

Εισαγωγή στην Python

Μεγάλοι ακέραιοι

Περιορίζονται μόνο από το μέγεθος της μνήμης

```
>>> 2**1000
10715086071862673209484250490600018105614048117055336074437503883703510511249361
22493198378815695858127594672917553146825187145285692314043598457757469857480393
45677748242309854210746050623711418779541821530464749835819412673987675591655439
46077062914571196477686542167660429831652624386837205668069376
>>> def fact(n): return 1 if n==1 else n*fact(n-1)
...
>>> fact(300)
30605751221644063603537046129726862938858880417357699941677674125947653317671686
74655152914224775733499391478887017263688642639077590031542268429279069745598412
25476930271954604008012215776252176854255965356903506788725264321896264299365204
57644883038890975394348962543605322598077652127082243763944912012867867536830571
22936819436499564604981664502277165001851765464693401122260347297240663332585835
06870150169794168850353752137554910289126407157154830282284937952636580145235233
15693648223343679925459409527682060806223281238738388081704960000000000000000000
00000000000000000000000000000000000000000000000000
```

Εισαγωγή στην Python

Λευκός χώρος (whitespace)

Ο λευκός χώρος του προγράμματος έχει σημασία για την Python, ειδικά οι εσοχές (indentation) και οι αλλαγές γραμμών (newlines)

- Μια αλλαγή γραμμής τερματίζει μια γραμμή κώδικα
  - Αν χρειάζεται "πρώιμη" αλλαγή γραμμής χρησιμοποιούμε το `\`
- Η ομαδοποίηση του κώδικα γίνεται με συνεπείς εσοχές
  - Δεν χρησιμοποιούμε `{` και `}` ή `begin` και `end`
  - Η μικρότερη εσοχή είναι έξω από την ομαδοποίηση
  - Η μεγαλύτερη εσοχή ξεκινά μια εμφωλευμένη (nested) ομαδοποίηση
  - Συνήθως εμφανίζεται ένα `:` πριν από τις εσοχές

```
def fibonacci(n):
    ''' This function prints the first n numbers of the fibonacci sequence '''
    a, b = 0, 1
    for i in range(n):
        print (a)
        a, b = b, a+b
```

Ερωτήσεις;