Προγραμματισμός Η/Υ

Περιεχόμενα

Τμήμα: Μηχανικών Χωροταξίας, Πολεοδομίας και Περιφερειακής Ανάπτυξης

Τίτλος Επιστημονικού Πεδίου: Πληροφορική και Ανάλυση Δεδομένων

Κωδικός Μαθήματος: ΜΕ0200

Τίτλος Μαθήματος: Προγραμματισμός Η/Υ

Κατηγορία Μαθήματος: Επιλογής

Εξάμηνο: Εαρινό

Περίγραμμα του μαθήματος

Το μάθημα εισάγει τους φοιτητές στις βασικές έννοιες και αρχές του προγραμματισμού και είναι προσαρμοσμένο στις ανάγκες και στο υπόβαθρο των χωροτακτών. Στόχος του μαθήματος είναι να αποκτήσουν οι φοιτητές τις αναγκαίες γνώσεις για την επεξεργασία και ανάλυση δεδομένων, την αυτοματοποίηση διαδικασιών και την σύνταξη σεναρίων (scripts) για την αναπαραγωγισιμότητα της ερευνάς τους. Πέρα από τις βασικές αρχές προγραμματισμού η διδασκαλία επεκτείνεται σε εξειδικευμένες θεματικές ενότητες που αφορούν τα γεωχωρικά δεδομένα και την ανάλυσή τους μέσω προγραμματισμού. Η διδασκαλία θα στηριχθεί στην Python, μια σύγχρονη, ευρέως διαδεδομένη και υψηλού επιπέδου γλώσσα προγραμματισμού. Επιπλέον, όπου κριθεί αναγκαίο, θα επιδειχτούν συμπληρωματικά διαδικασίες με την γλώσσα προγραμματισμού R.

Το πρόγραμμα των διαλέξεων καθώς και η προτεινόμενη βιβλιογραφία παρατίθενται στην συνέχεια.

1. Εισαγωγή στον προγραμματισμό

Ενότητες του μαθήματος

Το μάθημα χωρίζεται στις ακόλουθες ενότητες:

1. Εισαγωγή στον προγραμματισμό

Κατά την διάρκεια της διάλεξης διευκρινίζεται ο σκοπός του μαθήματος και περιγράφονται συνοπτικά οι ενότητες που θα διδαχθούν οι φοιτητές κατά την διάρκεια του εξαμήνου. Διατυπώνονται συγκεκριμένοι ορισμοί που αφορούν τον προγραμματισμό Η/Υ και αναπτύσσονται έννοιες για την επιστήμη των υπολογιστών. Στην συνέχεια εγκαθίσταται στους υπολογιστές των φοιτητών η γλώσσα προγραμματισμού Python μαζί με το απαραίτητο λογισμικό για την συγγραφή και αποσφαλμάτωση του κώδικα. Ακολουθεί εξοικείωση με το περιβάλλον εργασίας.

2. Τιμές, τύποι και μεταβλητές

Περιγραφή της έννοιας των μεταβλητών, των σταθερών, τύποι δεδομένων, εκχώρηση τιμών στις μεταβλητές, κανόνες ονοματοδοσίας των μεταβλητών.

3. Εκφράσεις, τελεστές

Ορισμός εκφράσεων, τι είναι τελεστές, ποια είναι η προτεραιότητα των τελεστών, πως εισάγουμε σχόλια στον κώδικα και γιατί είναι σημαντική πρακτική.

4. Έλεγχος ροής εκτέλεσης

Η λογική Boolean, Εκτέλεση υπό συνθήκη, αλυσιδωτές και εμφωλευμένες συνθήκες, βρόχος και οι εντολές επανάληψης for και while.

5. Συναρτήσεις

Ορισμός και κλήση συνάρτησης, παράμετροι συναρτήσεων, εμβέλεια μεταβλητών, αναδρομή.

6. Συμβολοσειρές/Δομές Δεδομένων

Προσπέλαση συμβολοσειρών, χαρακτήρες διαφυγής, υποσύνολα συμβολοσειράς, συγκρίσεις και ιδιότητες, μέθοδοι συμβολοσειρών. Λίστες, Πλειάδες, Λεξικά.

7. Ανάγνωση & εγγραφή αρχείων, φάκελοι

Ανάγνωση και εγγραφή σε αρχείο, σειριοποίηση (serialization) αντικειμένου, διαχείριση φακέλων και αρχείων.

8. Πίνακες και διαγράμματα

Ανάγνωση αρχείων csv ή excel, pandas dataframes

9. Πίνακες και διαγράμματα

Πίνακες στην βιβλιοθήκη numpy, διαγράμματα με την βιβλιοθήκη seaborn.

10. Γεωεπεξεργασία διανυσματικών δεδομένων

Ανάγνωση και εγγραφή διανυσματικών δεδομένων, μετα-δεδομένα, φιλτράρισμα, αλλαγή προβολικού συστήματος.

11. Ανάλυση διανυσματικών δεδομένων

Χωρικές σχέσεις, στατιστικά ομαδοποιήσεων, οπτικοποίηση διανυσματικών δεδομένων.

12. Γεωεπεξεργασία ψηφιδωτών δεδομένων

Ανάγνωση και εγγραφή διανυσματικών δεδομένων, μετα-δεδομένα, ορισμός μάσκας/αποκοπή περιοχής, αλλαγή τιμών, επαναταξινόμηση, αλλαγή προβολικού συστήματος.

13. Ανάλυση ψηφιδωτών δεδομένων

Άλγεβρα ψηφιδωτών αρχείων, στατιστικά ζωνών, ιστόγραμμα συχνοτήτων.

Ορισμοί

> **Definition 1**
>
> «*Αλγόριθμο*» ονομάζουμε κάθε πεπερασμένη και αυστηρά καθορισμένη σειρά βημάτων (οδηγιών) για την επίλυση ενός προβλήματος. [Αγγελιδάκης, 2015]
>
> Ένας αλγόριθμος είναι μια αυστηρά καθορισμένη διαδικασία που λαμβάνει μια τιμή ή ένα σύνολο τιμών εισόδου και αποδίδει μια ή περισσότερες τιμές εξόδου. Είναι κατά συνέπεια μια ακολουθία υπολογιστικών βημάτων που μετατρέπει την είσοδο δεδομένων σε έξοδο αποτελεσμάτων [Cormen, 2009].

Για παράδειγμα η αύξουσα (ή φθίνουσα) ταξινόμηση μιας λίστας αριθμών είναι ένα χαρακτηριστικό παράδειγμα αλγορίθμου. Οπότε με τιμές εισόδου {31, 41, 59, 26, 41, 58}, ο αλγόριθμος ταξινόμησης επιστρέφει ως τιμές εξόδου {26, 31, 41, 41, 58, 59}.

> **Definition 2**
>
> Ως «*Πρόγραμμα*» ορίζεται ένας αλγόριθμος γραμμένο σε γλώσσα κατανοητή για τον υπολογιστή και περιέχει εντολές (οδηγίες) που κατευθύνουν με κάθε λεπτομέρεια τον υπολογιστή, για να εκτελέσει μια συγκεκριμένη εργασία και να επιλύσει ένα πρόβλημα [Αγγελιδάκης, 2015]. Ένα Πρόγραμμα αναγνώσιμο από τον άνθρωπο ονομάζεται «*πηγαίος κώδικας*».

> **Definition 3**
>
> «*Προγραμματισμός*» ονομάζεται η διαδικασία συγγραφής προγραμμάτων και περιλαμβάνει τη διατύπωση των κατάλληλων εντολών προς τον υπολογιστή με τη χρήση τεχνητών γλωσσών, των γλωσσών προγραμματισμού [Αγγελιδάκης, 2015].

Ο προγραμματισμός είναι μια διαδικασία που απαιτεί μια σειρά εργαλείων και διαδικασιών τα οποία συνήθως ενσωματώνονται όλα μαζί σε ένα ενοποιημένο περιβάλλον που αποκαλείται «ολοκληρωμένο περιβάλλον ανάπτυξης εφαρμογών» (Integrated Development Environment, IDE).

Η διαδικασία μετατροπής του πηγαίου κώδικα σε εκτελέσιμο αρχείο περιγράφεται στο παρακάτω διάγραμμα:

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-1-00b6ee57d034> in <module>
----> 1 import graphviz
      2
      3 f = graphviz.Digraph('finite_state_machine', filename='fsm.gv')
      4 f.attr(rankdir='LR', size='8,5')
      5

ModuleNotFoundError: No module named 'graphviz'
```

Εικ. 1 Η ροή εκτέλεσης του κώδικα σε εκτελέσιμο.

Τα εργαλεία προγραμματισμού τα οποία κάνουν την μεταγλωττιστή του πηγαίου προγράμματος σε εκτελέσιμο πρόγραμμα είναι τα εξής:

- ο *επεξεργαστής κειμένου* με την βοήθεια οποίου συντάσσεται ο πηγαίο κώδικάς του προγράμματος.
- ο *μεταγλωττιστής* ή *διερμηνευτής* οι οποίοι χρησιμοποιούνται για την μετατροπή του πηγαίου κώδικα σε γλώσσα μηχανής η οποία είναι απαραίτητη για την αναγνώριση και εκτέλεση των εντολών από τον Η/Υ. Τα παραγόμενο πρόγραμμα από την μεταγλώττιση ονομάζεται *αντικείμενο πρόγραμμα* (object). Ο διερμηνευτής διαβάζει διαδοχικά τις εντολές και για κάθε εντολή που διαβάζει, εκτελεί αμέσως μια ισοδύναμη ακολουθία εντολών μηχανής. Από την άλλη, ο μταγλωττιστής δέχεται στην είσοδο ένα πρόγραμμα γραμμένο σε γλώσσα υψηλού επιπέδου (πηγαίο κώδικα) και παράγει ισοδύναμο πρόγραμμα σε γλώσσα μηχανής (αντικείμενο).
- ο *συνδέτης - φορτωτής* (linker- loader) ο οποίος συνδέει το *αντικείμενο πρόγραμμα* με άλλα τμήματα του προγράμματος ή απαραίτητες βιβλιοθήκες που διατίθεται από την γλώσσα προγραμματισμού. Το τελικό πρόγραμμα που προκύπτει από την μεταγλώτισση και την σύνδεση των τμημάτων του προγράμματος είναι το *εκτελέσιμο πρόγραμμα* (executable) το οποίο μπορεί να διαβάσει και να εκτελέσει ο υπολογιστής.
- τα *εργαλεία αποσφαλμάτωσης* με την βοήθεια των οποίων δοκιμάζεται η εκτέλεση και η ορθότητα του πηγαίου κώδικα και εντοπίζονται λάθη σε αυτόν.

Τα λάθη στον κώδικα συνοψίζονται σε τρείς βασικές κατηγορίες:

1. *σφάλμα μεταγλώττισης* τα οποία προκύπτουν κατά την λανθασμένη συγγραφή του πηγαίου κώδικα. Ο μεταγλωττιστής δεν επιτρέπει την μετάφραση του πηγαίου κώδικα σε γλώσσα μηχανής αν προηγουμένος δεν έχει διορθωθεί το συντακτικό λάθος. Συντακτικά λάθη συμβαίνουν συνήθως όταν δεν ακολοθούνται οι κανόνες σύνταξης μια γλώσσας (π.χ. μια παρένθεση που δεν έχει κλείσει, ένα ξεχασμένο εισαγωγικό ή κόμμα κτλ.).

Το παρακάτω είναι ένα παράδειγμα συντακτικού σφάλματος και το μήνυμα που επιστρέφει ο μεταγλωττιστής της Python. Η αιτία του σφάλματος είναι η ξεχασμένη παρένθεση στην συνάρτηση (function) *print*

```
print("an example"
```

1. *σφάλμα εκτέλεσης* (run-time errors) τα οποία συμβαίνουν κατά την εκτέλεση του προγράμματος παρότι δεν υπάρχουν σφάλματα σύνταξης. Χαρακτηριστικά παραδείγματα τέτοιων λαθών είναι η διαίρεση με το μηδέν, η πρόσβαση σε ένα στοιχείο μιας λίστας εκτός του εύρους της, η ανάγνωση ενός αρχείου το οποίο δεν υπάρχει, ή η πρόσβαση σε ένα ανύπαρκτο object. Τα σφάλματα εκτέλεσης έχουν επικρατήσει να αναφέρονται και ως «bugs» [1]. Παρακάτω δίνεται ένα σφάλμα που προκύπτει από την διαίρεση ενός ακέραιου με το μηδέν.

```
1/0
```

1. *σφάλμα λογικής*, κατά το οποίο το πρόγραμμα εκτελείται κανονικά χωρίς σφάλματα αλλά δεν συμπεριφέρεται όπως έχει σχεδιαστεί να συμπεριφέρεται. Αυτά τα σφάλματα δεν σταματούν την εκτέλεση του προγράμματος αλλά το αποτέλεσμα της εκτέλεσης δεν ειναι το αναμενόμενο.

```
x = 6
y = 4

z = x+y/2
print('Ο μέσος όρος των δύο αριθμών είναι:',z)
```

Το παραπάνω είναι σφάλμα λογικής γιατί έπρεπε να γραφτεί ως εξής (δώστε προσοχή στις παρενθέσεις που δίνουν προτεραιότητα στις πράξεις):

```
x = 6
y = 4

z = (x+y)/2
print('Ο μέσος όρος των δύο αριθμών είναι:',z)
```

Όλες οι παραπάνω μορφές σφαλμάτων εντοπίζονται μέσω της *αποσφαλμάτωσης*, της συστηματικής δηλαδή διαδικασίας εντοπισμού και επιδιόρθωσης σφαλμάτων. Η αποσφαλμάτωση συνοψίζεται στα εξής βήματα:

- Επανάληψη του προβλήματος
- Απομόνωση του σημείου που εμφανίζεται το σφάλμα
- Αναγνώριση της αιτίας που το προκαλεί
- Διόρθωση του σφάλματος
- Επιβεβαίωση της διόρθωσης

Οι εντολές των προγραμμάτων γράφονται από τους προγραμματιστές σε τεχνητές γλώσσες που ονομάζονται «*γλώσσες προγραμματισμού*». Μια γλώσσα προγραμματισμού θα πρέπει να έχει αυστηρά ορισμένη σύνταξη και σημασιολογία. Η σύνταξη καθορίζει αν μια σειρά από σύμβολα αποτελούν «νόμιμες» εντολές ενός προγράμματος γραμμένου σε μια συγκεκριμένη γλώσσα προγραμματισμού και η σημασιολογία καθορίζει τη σημασία του προγράμματος, δηλαδή τις υπολογιστικές διαδικασίες που υλοποιεί. [Αγγελιδάκης, 2015].

Η γλώσσα προγραμματισμού Python

Η Python είναι μια ευρέως διαδομένη, αντικειμενοστραφής, υψηλού επιπέδου γλώσσα προγραμματισμού γενικής χρήσης. Η Python είναι μια γλώσσα που εκτελεί τις εντολές στον διερμηνέα, που όπως αναφέρθηκε,διαβάζει τον πηγαίο κώδικα γραμμή προς γραμμή και το μετατρέπει σε γλώσσα μηχανής. Αυτός ο τρόπος λειτουργίας της Python την καθιστά πιο αργή σε σύγκριση με άλλες γλώσσες μεταγλωττιστού όπως η C. Η Python είναι διαδραστική υπό την έννοια ότι ο χρήστης εκτελεί εντολές μέσω της γραμμή εντολών της Python, εκτελείται άμεσα και λαμβάνει το αποτέλεσμα εξόδου.

Δημιουργήθηκε από τον Guido van Rossum και πρωτοκυκλοφόρησε στις 20 Φεβρουαρίου του 1991. Το όνομά της, αν και παρεπέμπει, δεν έχει σχέση με το φίδι Πύθωνα αλλά προέρχεται από την γνωστή κωμικη σειρά του BBC, Monty Python's Flying Circus. Αν και αρχικά αναπτύχθηκε σαν μεμονωμένη ατομική προσπάθεια στην συνέχεια υποστηρίχθηκε από μια παγκόσμια κοινότητα προγραμματιστών και χρηστών. Στις 6 Μαρτίου 2001 ιδρύθηκε το αμερικάνικο μη κερδοσκοπικό ίδρυμα *Python Software Foundation (PSF)*, το οποίο στόχο έχει την διάδοση και υποστήριξη της Python μέσω της διοργάνωσης συνεδρίων, την ανάπτυξη κοινοτήτων χρηστών, την υποστήριξη προσπαθειών μέσω υποτροφιών και την διασφάλιση οικονομικών πόρων για την ανάπτυξη της γλώσσας. Το ίδρυμα κατέχει τα πνευματικά δικαιώματα της γλώσσας και διασφαλίζει ότι αυτή θα διατίθεται με όρους ελεύθερου λογισμικού προς το ευρύτερο κοινό.

Οι βασικοί στόχοι που έθεσε ο δημιουργός κατά την ανάπτυξή της είναι να είναι εύκολη και κατανοητή με ισχυρές δυνατότητες εφάμιλλες των ανταγωνιστικών γλωσσών. Ταυτόχρονα έθεσε το όρο να είναι ανοιχτού κώδικα (open source) για να μπορεί εύκολα να αναπτύσσεται απο τους ενδιαφερόμενους προγραμματιστες και να έχει πρακτική αξία σε καθημερινές εργασίες ρουτίνας. Την τρέχουσα περίοδο (03/2022) κατατάσσεται ως η κορυφαία γλώσσα προγραμματισμού σύμφωνα με την κοινότητα προγραμματιστών TIOBE αλλά και τον δείκτη PopularitY of Programming Language Index (PYPL).

Mar 2022	Mar 2021	Change	Programming Language	Ratings	Change
1	3	↑	Python	14.26%	+3.95%
2	1	↓	C	13.06%	-2.27%
3	2	↓	Java	11.19%	+0.74%
4	4		C++	8.66%	+2.14%
5	5		C#	5.92%	+0.95%
6	6		Visual Basic	5.77%	+0.91%
7	7		JavaScript	2.09%	-0.03%
8	8		PHP	1.92%	-0.15%
9	9		Assembly language	1.90%	-0.07%
10	10		SQL	1.85%	-0.02%
11	13	↑	R	1.37%	+0.12%
12	14	↑	Delphi/Object Pascal	1.12%	-0.07%
13	11	↓	Go	0.98%	-0.33%
14	19	⇈	Swift	0.90%	-0.05%
15	18	↑	MATLAB	0.80%	-0.23%
16	16		Ruby	0.66%	-0.52%
17	12	⇊	Classic Visual Basic	0.60%	-0.66%
18	20	↑	Objective-C	0.59%	-0.31%
19	17	↓	Perl	0.57%	-0.58%
20	38	⇈	Lua	0.56%	+0.23%

Εικ. 2 Η κατάταξη σύμφωνα με την κοινότητα TIOBE (Μάρτιος 2022)

Η Python πλέον είναι μια ώριμη γλώσσα προγραμματισμού με εφαρμογές στην ανάπτυξη διαδικτυακών εφαρμογών και υπηρεσιών, την εκπαίδευση, την ανάλυση δεδομένων, την τηλεπισκόπηση και τα ΣΓΠ, την δημιουργία γραφικών, την διαχείριση συστημάτων, τα παιχνίδια, το εμπόριο και την επιχειρηματικότητα, τους μικροελεγκτές και το Internet of Things (IOT).

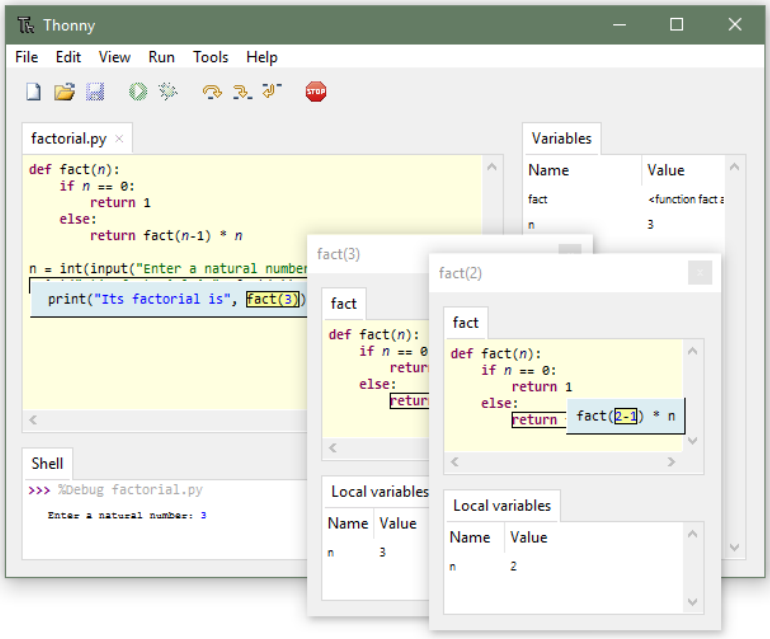
Η φιλοσοφία της Python ως προς την μεθοδολογία ανάπτυξης και προγραμματισμού συνοψίζεται σε 20 αρχές, οι οποίες εκτυπώνονται μέσω της γλώσσας με την παρακάτω εντολή:

```
import this
```

Το ολοκληρωμένο περιβάλλον ανάπτυξης thonny

Η συγγραφή κώδικα θα γίνει στο ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment, IDE) thonny. Δεν απαιτείται η προεγκατάσταση της Python καθώς το thonny έρχεται με ενσωματωμένη την γλώσσα προγραμματισμού Python 3.7 και διατίθεται για Windows, Mac και Linux. Το thonny αποτελεί εκπαιδευτικό περιβάλλον για την συγγραφή και αποσφαλμάτωση κώδικα Python. Για τον λόγο αυτό διαθέτει πολύ συγκεκριμένες αλλά ζωτικές λειτουργίες και δεν είναι επιφορτισμένο με δυνατότητες που απαιτούνται από προχωρημένους προγραμματιστές. Το Γραφικό Περιβάλλον Χρήστη (GUI, Graphical User Interface) είναι λιτό ώστε να μην αποπροσανατολίζει τον αρχάριο χρήστη. Το thonny παρέχει βοηθητικές λειτουργίες για τον χρήστη όπως είναι η σήμανση συντακτικών λαθών, η αυτόματη συμπλήρωση κώδικα και η ευκολία στην επέκταση των λειτουργίων της Python με την εγκατάσταση συμπληρωματικών πακέτων.

Ενναλλακτικά, στους χρήστες παρέχεται online περιβάλλον ανάπτυξης που βασίζεται στο Jupyter Lab. Η χρήση του δεν απαιτεί την εγκατάσταση λογισμικού παρά μόνον έναν απλό φυλλομετρητή (προτείνεται Chrome, Safari ή Firefox). Το online περιβάλλον Jupyter είναι προσβάσιμο από εδώ: https://kokkytos.github.io/programming



Εικ. 3 Το περιβάλλον εργασία thonny.

Εκτέλεση εντολών στο περιβάλλον thonny

Στην παρακάτω ενότητα παρουσιάζονται μερικά εισαγωγικά παραδείγματα από εντολές της Python. Δεν θα επικεντρωθούμε σε λεπτομέρειες ούτε θα αναλύσουμε τις εντολές που διατυπώνονται στα αρχεία. Παρουσιάζονται σαν μια μορφή συνοπτικής επίδειξης των δυνατοτήτων που διαθέτει η γλώσσα και θα περιγράψουμε σε επόμενα μαθήματα.

Δοκιμάστε να τρέξετε την παρακάτω εντολή στην γραμμή εντολών της Python στο thonny:

```
25+30
```

Τι παρατηρείτε; Η Python λειτούργησε σαν μια απλή αριθμομηχανή.

Στην συνέχεια δοκιμάστε να τρέξετε την παρακάτω εντολή γραμμή προς γραμμή:

```
number1 = 25
number2 = 30
number3 = number1+number2
number3
```

Το αποτέλεσμα είναι το ίδιο με το προηγούμενο.

Δώστε την παρακάτω εντολή. Αντικαταστήστε την συμβολοσειρά *AnyName* με το ονομά σας.

```
name="AnyName"

for i in range(10):
    print("Εκτύπωση", i,":",  name)
```

Όπως βλέπετε η Python επανέλαβε την εκτύπωση του ονόματός σας 10 φορές. Από που ξεκινά όμως η αρίθμηση της πρώτης εκτύπωσης;

Στο επόμενο παράδειγμα η Python θα σας ενημερώσει αν τρέχετε γρήγορα ή αργά ή αν είστε ακίνητος:

```
speed=70

if speed>50:
    if speed>=100:
        print("Τρέχεις πολύ γρήγορα")
    else:
        print("Τρέχεις γρήγορα")
else:
    if speed==0:
        print("Είσαι ακίνητος".upper())
    if speed>0:
        print("Τρέχεις αργά")
```

Έστω ότι κινείσθε σε αυτοκινητόδρομο με 120 km/h. Αν ορίσετε την ταχύτητα (speed) στον κώδικα, τι θα σας απαντήσει η Python; Αν κινείστε με μηδενική ταχύτητα (speed=0) τι μήνυμα θα λάβετε; Υπάρχουν περιπτώσεις που η Python, και δικαιολογημένα, αγνοεί να απαντήσει με μήνυμα στην ταχύτητα που ορίζεται. Μπορείτε να εντοπίσετε σε ποιές περιπτώσεις;

---

[1] Η πρώτη περίπτωση *bug* σε υπολογιστή καταγράφεται το 1947 από τον Grace Murray Hopper και πρόκειται για την κυριολεκτική έννοια του όρου. Στο ημερολόγιό του καταγράφει προβλήματα στην λειτουργία του υπολογιστή του Harvard, Mark II, από την ύπαρξη ενός εντόμου στο εσωτερικό του κύκλωμα.

Βιβλιογραφία

1 Νικόλαος Αγγελιδάκης. *Εισαγωγή στον προγραμματισμό με την Python*. Αγγελιδάκης, Ηράκλειο, 2015. ISBN 978-960-93-7364-7.

2 Thomas Cormen, editor. *Introduction to algorithms*. MIT Press, Cambridge, Mass, 3rd ed edition, 2009. ISBN 978-0-262-03384-8 978-0-262-53305-8. OCLC: ocn311310321.

Αξιολόγηση

Η αξιολόγηση γίνεται:

- 50% από ατομικές ασκήσεις στην διάρκεια του εξαμήνου.
- 50% από τελικές εξετάσεις.

Με Λεωνίδας Λιάκος