Ενότητα 2

Προγραμματισμος με τη Γλωσσα Python

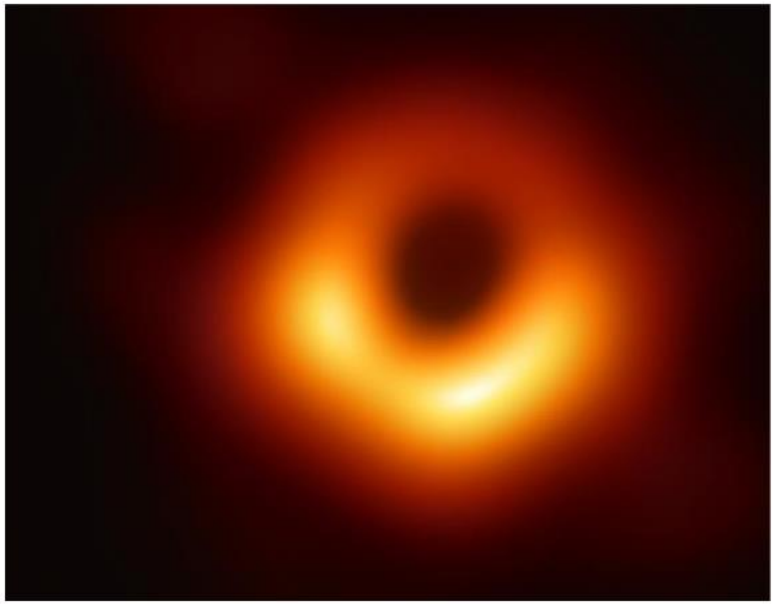
Το περιβάλλον EduBlocks

Η γλώσσα Python

Ανάπτυξη εφαρμογών

Καθημερινά χρησιμοποιούμε διάφορες εφαρμογές, οι οποίες μας διευκολύνουν σε πολλές εργασίες. Αν θέλουμε να βρούμε πληροφορίες για κάτι που δε γνωρίζουμε χρησιμοποιούμε μια μηχανή αναζήτησης, όπως η **Google** ή ένα γλωσσικό μοντέλο Τεχνητής Νοημοσύνης, όπως το **ChatGPT**. Αν θέλουμε να αποθηκεύσουμε τα αρχεία μας σε έναν δικτυακό φάκελο στο υπολογιστικό νέφος, ώστε να έχουμε πρόσβαση από παντού στα αρχεία μας, χρησιμοποιούμε μια εφαρμογή διαμοιρασμού αρχείων, όπως είναι το **DropBox**. Για ψυχαγωγία χρησιμοποιούμε εφαρμογές, όπως το **Spotify** για μουσική ή το **Instagram** για επικοινωνία ή το **Netflix**, για να δούμε ταινίες ή το youtube, για να δούμε ή να ανεβάσουμε βίντεο.

Όλες αυτές οι εφαρμογές έχουν κάτι κοινό. Είναι γραμμένες στην ίδια γλώσσα προγραμματισμού, τη γλώσσα Python. Η Python είναι μια υψηλού επιπέδου γλώσσα προγραμματισμού, η οποία δημιουργήθηκε από τον Ολλανδό Guido van Rossum το 1990. Η γλώσσα Python χρησιμοποιείται και σε επιστημονικές εφαρμογές, όπως για παράδειγμα στην αστροφυσική, όπου οι επιστήμονες οπτικοποίησαν για πρώτη φορά μια μαύρη τρύπα.

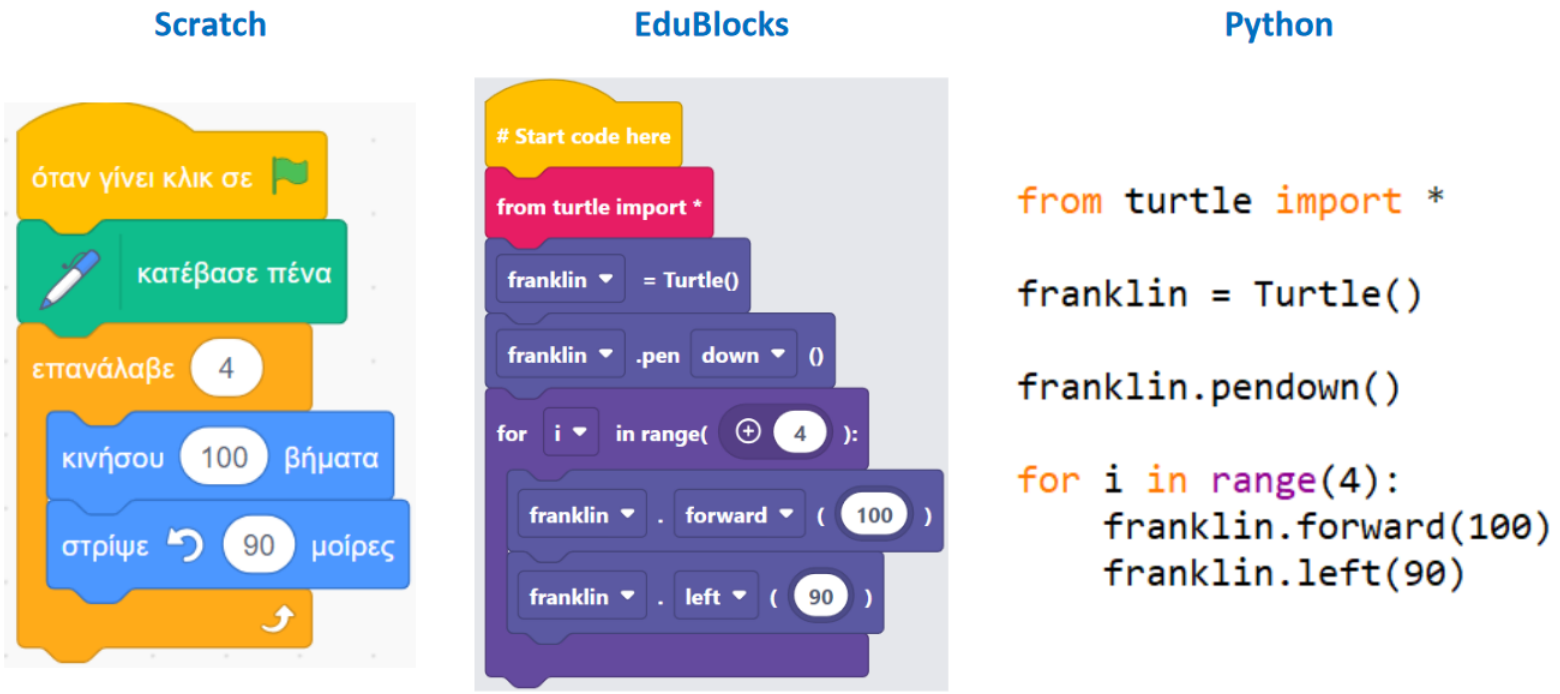


Εικόνα 2.1. Η πρώτη εικόνα μαύρης τρύπας (M87) που δημιουργήθηκε από λογισμικό Τεχνητής Νοημοσύνης γραμμένο στη γλώσσα Python από την ερευνητική ομάδα του Event Horizon Telescope

Εικόνα 2.2. Η ερευνήτρια Katie Bouman μόλις βλέπει για πρώτη φορά το αποτέλεσμα του αλγορίθμου οπτικοποίησης της μαύρης τρύπας που ανέπτυξε η ίδια στη γλώσσα Python

Σε αυτήν την ενότητα θα γράψετε τα πρώτα σας προγράμματα σε μια **πραγματική γλώσσα προγραμματισμού** η οποία χρησιμοποιείται ευρέως από προγραμματιστές, μηχανικούς και επιστήμονες για την ανάπτυξη χρήσιμων εφαρμογών.

Η Python είναι μια υψηλού επιπέδου γλώσσα προγραμματισμού η οποία δημιουργήθηκε από τον Ολλανδό Guido van Rossum το 1990. Ένα σημαντικό πλεονέκτημά της είναι η ευκολία χρήσης της. Για τον λόγο αυτό χρησιμοποιείται από πολλά πανεπιστήμια και σχολεία σε εισαγωγικά μαθήματα αλγοριθμικής.



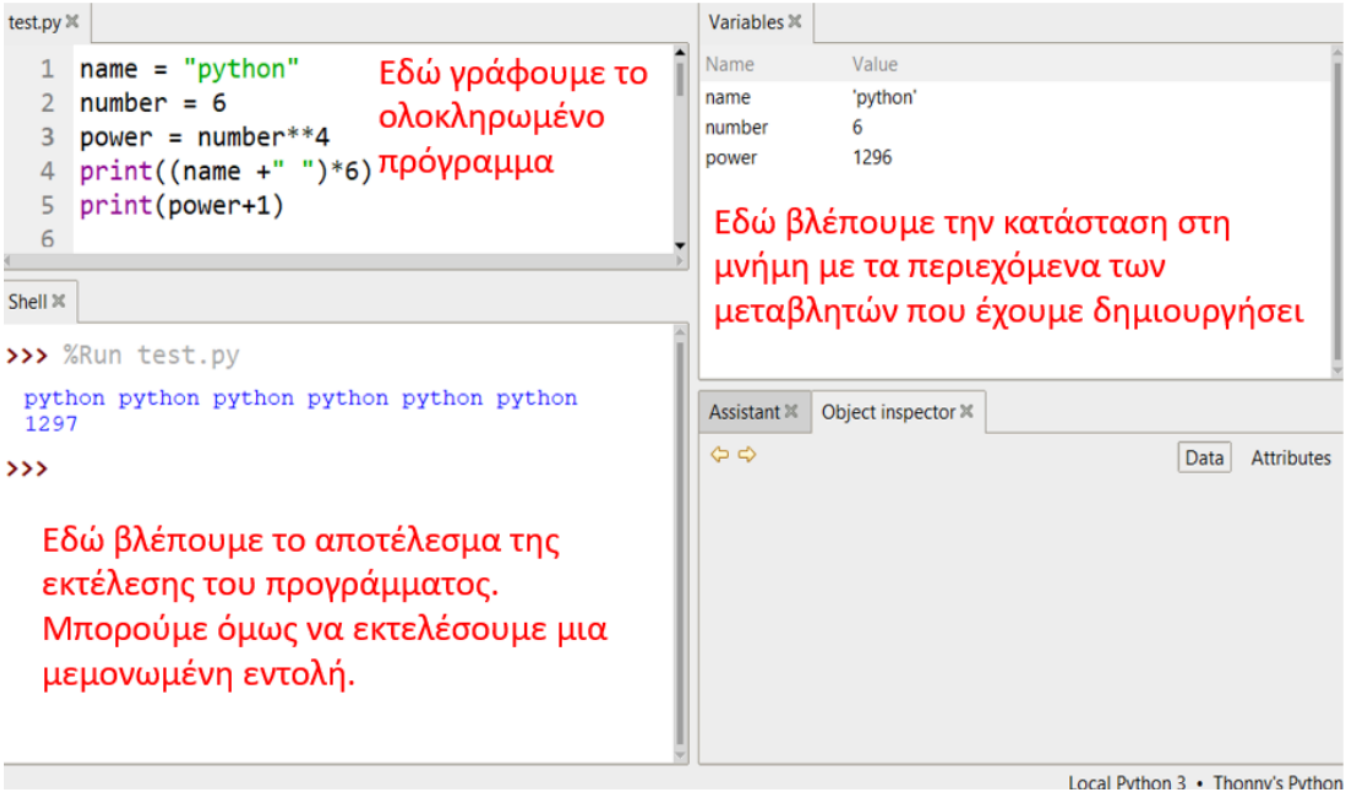
Εικόνα 2.3. Αλγόριθμος για τον σχεδιασμό τετραγώνου σε διαφορετικά περιβάλλοντα προγραμματισμού

Παραπάνω φαίνεται ο σχεδιασμός ενός τετραγώνου με πλευρά 100 σε Scratch, σε Edublocks και στη γλώσσα Python. Στο περιβάλλον Edublocks κάθε μπλοκ αντιστοιχεί σε μια εντολή της γλώσσας Python. Η διαφορά με το Scratch είναι ότι εδώ θα πρέπει πρώτα να δημιουργήσουμε το αντικείμενο, π.χ. τον franklin το χελωνάκι, στο οποίο θα δώσουμε τις κατάλληλες εντολές, για παράδειγμα να κινηθεί μπροστά (franklin.forward()) ή να στρίψει αριστερά (franklin.left()). Αυτού του είδους ο προγραμματισμός που είναι προσανατολισμένος ή, όπως λέγεται, στρέφεται προς το αντικείμενο λέγεται **αντικειμενοστρεφής.** Παρακάτω δίνεται μια εξήγηση για καθεμία από τις εντολές της Python που εμφανίζονται στο παραπάνω πρόγραμμα.

from turtle import *	Προσάρτηση της βιβλιοθήκης turtle την οποία θα χρησιμοποιήσουμε.
franklin = Turtle( )	Δημιουργία ενός αντικειμένου με όνομα franklin, τύπου Turtle.
franklin.pendown( )	Εντολή στον franklin να κατεβάσει την πένα, ώστε να αφήνει το ίχνος του.
franklin.forward(100)	Εντολή στον franklin να προχωρήσει μπροστά.
franklin.left(90)	Εντολή στον franklin να στρίψει αριστερά 90^{o}.
for i in range(4):	Επανάλαβε 4 φορές

2.2.2 Το περιβάλλον προγραμματισμού Thonny

Το Thonny είναι ένα δωρεάν και ανοικτού κώδικα περιβάλλον προγραμματισμού για τη γλώσσα Python. Μπορείτε να το κατεβάσετε από τη διεύθυνση: https://thonny.org/.



Εικόνα 2.6. Το περιβάλλον προγραμματισμού Thonny

Εκτός από το παράθυρο του συντάκτη κειμένου που βρίσκεται πάνω αριστερά μπορούμε να δοκιμάσουμε και εντολές στο τερματικό του διερμηνευτή, αν θέλουμε να πειραματιστούμε και να δούμε τι λειτουργία επιτελούν.

Σχεδίασε ένα τετράγωνο

Σχεδίασε ένα σπιράλ

Τύποι δεδομένων

Κάθε γλώσσα προγραμματισμού αποθηκεύει τα δεδομένα στη μνήμη σε διάφορες μορφές. Στο βιβλίο αυτό θα διαχωρίσουμε τα δεδομένα σε δυο βασικές κατηγορίες: τις λέξεις και τους αριθμούς.

Οι λέξεις στην ορολογία της Πληροφορικής λέγονται αλφαριθμητικά, επειδή μπορούν να περιέχουν γράμματα και αριθμούς, όπως για παράδειγμα ο κωδικός πρόσβασής σας (password) στην ηλεκτρονική τάξη. Τα **αλφαριθμητικά** είναι μια ακολουθία από χαρακτήρες που μπορούν να είναι ψηφία, γράμματα ή σημεία στίξεως και βρίσκονται μέσα σε εισαγωγικά (quotes) (διπλά ή μονά).

Οι **αριθμητικοί τύποι** στην Python είναι οι ακέραιοι (integer) (**int**) και οι πραγματικοί (**float**) αριθμοί. Για την υποδιαστολή στον προγραμματισμό χρησιμοποιείται η τελεία «.» και όχι το κόμμα.

Τα δεδομένα μπορεί να είναι αριθμοί, ονόματα ή αποτελέσματα λογικών αποφάσεων. Στην Python έχουμε τους εξής βασικούς τύπους:

Τύπος	Ονομασία	Παραδείγματα
Ακέραιοι	int	1, 0, –1, 496
Πραγματικοί	float	3.14159, 0.5, 3.0
Λογικές	bool	True, False
Αλφαριθμητικά	str	“SARS-CoV-2”, “Ζάννειο Γυμνάσιο”
Λίστες	list	[6, 28, 496], [“Γη”, “Σελήνη”, “Ήλιος”]

Οι **λίστες** είναι ο σημαντικότερος σύνθετος τύπος της Python. Μια λίστα είναι μια ακολουθία από αντικείμενα οποιουδήποτε τύπου. Οι λίστες είναι δυναμικές δομές, δηλαδή μπορούμε να προσθέσουμε ή να αφαιρέσουμε στοιχεία αυξομειώνοντας το μέγεθός τους.

Για να ελέγξουμε τον τύπο δεδομένων χρησιμοποιούμε την εντολή type ().

```
>>> type(3)
<class 'int'>
>>> type(2.5)
<class 'float'>
>>> type('Hello, World!')
<class 'str'>
>>>
```

Δοκιμάστε στον διερμηνευτή …

Φύλλο εργασίας 1
Ασκ. 1

Η Python μας δίνει τη δυνατότητα να εκτελέσουμε διάφορες πράξεις στα δεδομένα, χρησιμοποιώντας τους αντίστοιχους τελεστές. Οι τελεστές (operators) είναι σύμβολα ή λέξεις για τη δημιουργία αριθμητικών και λογικών εκφράσεων. Οι βασικότερες κατηγορίες τελεστών είναι οι αριθμητικοί, οι συγκριτικοί και οι λογικοί. Υπάρχουν όμως και τελεστές για πράξεις σε αλφαριθμητικά δεδομένα και σε λίστες αντίστοιχοι με τους αριθμητικούς τελεστές.

Αριθμητικοί τελεστές: είναι τα σύμβολα που χρησιμοποιούμε για να κάνουμε μαθηματικές πράξεις. Ο υπολογισμός κάθε έκφραση στην οποία υπάρχουν αριθμητικοί τελεστές ακολουθεί μια αυστηρά καθορισμένη ιεραρχία πράξεων, που είναι:

1. Ύψωση σε δύναμη.
2. Πολλαπλασιασμός, διαίρεση.
3. Πρόσθεση, αφαίρεση.

Δοκιμάστε στον διερμηνευτή …

Φύλλο εργασίας 1
Ασκ. 2

Αριθμητικοί τελεστές: είναι τα σύμβολα που χρησιμοποιούμε για να κάνουμε μαθηματικές πράξεις. Ο υπολογισμός κάθε έκφραση στην οποία υπάρχουν αριθμητικοί τελεστές ακολουθεί μια αυστηρά καθορισμένη ιεραρχία πράξεων, που είναι: **1.** Ύψωση σε δύναμη. **2.** Πολλαπλασιασμός, διαίρεση. **3.** Πρόσθεση, αφαίρεση.	Πρόσθεση Αφαίρεση Πολλαπλασιασμός Διαίρεση Πηλίκο Ακέραιας Διαίρεσης Ύψωση σε δύναμη Υπόλοιπο ακέραιας διαίρεσης	+ – * / // ** %

Αν θέλουμε να αλλάξουμε την ιεραρχία των πράξεων, μπορούμε να χρησιμοποιήσουμε παρενθέσεις. Για παράδειγμα, στην έκφραση (200+1)*10 θα εκτελεστεί πρώτα η πρόσθεση μέσα στην παρένθεση και μετά το αποτέλεσμα θα πολλαπλασιαστεί επί 10, που μας κάνει 2010, σε αντίθεση με την έκφραση 200+1*10, στην οποία πρώτα θα γίνει ο πολλαπλασιασμός και μετά η πρόσθεση, άρα θα πάρουμε 210.

Ιεραρχία των Πράξεων

Η έκφραση

3 + 4 * 8 - (3 + 2)

Μπορεί να οδηγήσει σε διάφορα αποτελέσματα ανάλογα με το ποιες από τις πράξεις που διαθέτει θα εκτελεστούν πρώτα.

Σειρά Πράξεων:

1.Πραγματοποιήστε πρώτα τις πράξεις σε παρενθέσεις, **2.** στη συνέχεια τις δυνάμεις, **3.** συνεχίστε με πολλαπλασιασμούς και διαιρέσεις με τη σειρά με την οποία θα τα αντιμετωπίσετε από δεξιά προς τα αριστερά, **4.** Τελευταίες έρχονται οι προσθέσεις και αφαιρέσεις. Εκτελέστε με τη σειρά που θα τις αντιμετωπίσετε από αριστερά προς τα δεξιά.

Αποτέλεσμα της παραπάνω έκφρασης:

30

Ακέραια Διαίρεση & Υπόλοιπο

Ο τελεστής της **Ακέραιας Διαίρεσης** να θυμίσουμε ότι είναι το σύμβολο **//**
ενώ ο τελεστής του **Ακέραιου Υπολοίπου** είναι το σύμβολο **%**

Η **Ακέραια Διαίρεση** είναι το ακέραιο μέρος της απλής διαίρεσης μεταξύ δύο αριθμών.
Π.χ.
15 // 2 ⇒ 7 και όχι 7.5 που θα έδινε η απλή διαίρεση.

Το **Ακέραιο Υπόλοιπο** είναι το υπόλοιπο που μένει από την Ακέραια Διαίρεση.
Π.χ.
15 % 2 ⇒ 1 διότι το 2 «χωράει» 7 ακέραιες φορές στο 15 και μένει υπόλοιπο 1.

Δοκιμάστε στον διερμηνευτή …

Φύλλο εργασίας 1
Ασκ. 3

Μπορούμε να δοκιμάσουμε εντολές ή να υπολογίσουμε εκφράσεις χρησιμοποιώντας τον διερμηνευτή της Python, ο οποίος μεταφράζει σε γλώσσα μηχανής και εκτελεί μια μεμονωμένη εντολή χωρίς να χρειαστεί να αναπτύξουμε ολόκληρο πρόγραμμα.

Ο τελεστής (συνάρτηση) **str** μετατρέπει ένα αντικείμενο άλλου τύπου σε αλφαριθμητικό.
Ο τελεστής **int** μετατρέπει ένα αντικείμενο σε ακέραιο αριθμό, εάν αυτό είναι εφικτό.
Ο τελεστής **float** μετατρέπει ένα αντικείμενο σε πραγματικό αριθμό, εάν αυτό είναι εφικτό.

Ο τελεστής της πρόσθεσης **«+»** όταν εφαρμόζεται σε αλφαριθμητικά, εκτελεί συνένωση
ενώ αυτός του πολλαπλασιασμού **«*»** παράγει τη συμβολοσειρά επαναλαμβανόμενη τόσες φορές, όσες είναι ο αριθμός.

Δοκιμάστε στον διερμηνευτή …

Φύλλο εργασίας 1
Ασκ. 4

Φέτα συμβολοσειράς

Για να λάβουμε μια υπο-συμβολοσειρά μιας συμβολοσειράς παίρνουμε, όπως λέμε, μία φέτα (slice) της. Για να γίνει αυτό, χρησιμοποιούμε τον τελεστή **[n:m]**, ο οποίος επιστρέφει το τμήμα της συμβολοσειράς από τον n-οστό χαρακτήρα μέχρι τον m-στό χαρακτήρα, περιλαμβάνοντας τον πρώτο αλλά αποκλείοντας τον τελευταίο.

Ας δούμε, για παράδειγμα, τη συμβολοσειρά 'Tom and Jerry':

Εντολή	Αποτέλεσμα
'Tom and Jerry‘[0:3]	‘Tom’
'Tom and Jerry‘[? : ?]	‘and’
'Tom and Jerry' [?: ?]	‘Jerry’
'Tom and Jerry‘[0:1]	??

Δοκιμάστε στον διερμηνευτή …

Φύλλο εργασίας 1
Ασκ. 5

η συνάρτηση **len** επιστρέφει το μήκος, δηλαδή το πλήθος των χαρακτήρων του αλφαριθμητικού

π.χ

```
>>> len('σκουληκομερμηγκότρυπα')
```

η συνάρτηση **str** μετατρέπει μια τιμή σε συμβολοσειρά