

Σημειώσεις Python B

Πληροφορική ΕΠΑΛ

Βασικά Στοιχεία προγραμματισμού

Τι είναι μια γλώσσα προγραμματισμού;

Γλώσσα προγραμματισμού είναι μια **τεχνητή** γλώσσα που μπορεί να χρησιμοποιηθεί για τον προγραμματισμό ενός υπολογιστή. Μια γλώσσα προγραμματισμού ορίζεται από ένα σύνολο **εντολών** (ή λεξεων) που υποστηρίζει καθώς και από τους **συντακτικούς** κανόνες με τους οποίους σχηματίζονται οι προτάσεις της γλώσσας.

Τι είναι η Python;

Η **Python** είναι μια **γλώσσα προγραμματισμού** με πολλές δυνατότητες, που μπορεί να χρησιμοποιηθεί τόσο για απλές όσο και για σύνθετες εφαρμογές υπολογιστών (παιχνίδια, ιστοσελίδες, ρομποτική, βάσεις δεδομένων κ.α.)

Ποια είναι τα χαρακτηριστικά της Python;

- ❖ Διαθέτει πλούσια **βιβλιοθήκη** που επεκτείνει τη λειτουργία της
- ❖ Είναι σχετικά εύκολη στην εκμάθηση και κατάλληλη για μαθητές σχολείου
- ❖ Έχει **απλή** και καθαρή σύνταξη
- ❖ Είναι γλώσσα **γενικής χρήσης**, έχει εφαρμογές σε πολλά πεδία δηλαδή
- ❖ Είναι **φορητή**, εκτελείται δηλαδή σε διαφορετικά υπολογιστικά συστήματα
- ❖ Είναι διαδραστική γιατί χρησιμοποιεί **διερμηνευτή** ως μεταφραστικό πρόγραμμα
- ❖ Υποστηρίζει **διαδικασιακό** αλλά και **αντικειμενοστραφή** προγραμματισμό

Τι είναι ο πηγαίος κώδικας και το πρόγραμμα

Ένα πρόγραμμα είναι το **σύνολο εντολών** που έχουν σκοπό την επίλυση ενός προβλήματος από έναν υπολογιστή. Όταν ο προγραμματιστής γράφει τις εντολές ενός προγράμματος σε μια γλώσσα προγραμματισμού τότε αυτό ονομάζεται **πηγαίος κώδικας** (source code).

```
name = input("Πως σε λένε; ")
year = int(input("Ποια χρονιά γεννήθηκες; "))
age = 2020 - year
print("Καλησπέρα", name)
print("Πρέπει να είσαι", age, "χρονών")
```

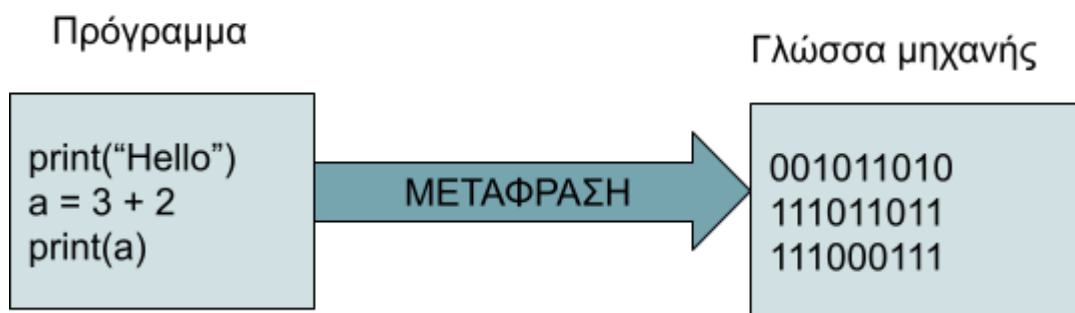
Τι είναι η γλώσσα μηχανής;

Ο πηγαίος κώδικας και το πρόγραμμα δεν είναι άμεσα κατανοητός από τον υπολογιστή, γιατί αυτός δε «μιλά» τη γλώσσα Python, αλλά τη γλώσσα που «μιλά» η μηχανή/υπολογιστής, τη λεγόμενη **γλώσσα μηχανής**. Η γλώσσα μηχανής αποτελείται από λέξεις που περιλαμβάνουν διαδοχικές σειρές μηδέν (0) και ένα (1), π.χ. 01100. Τότε και μόνον τότε, το πρόγραμμα θα εκτελεστεί από τον υπολογιστή και θα παράγει αποτελέσματα.

Τι είναι η μετάφραση προγράμματος

Για να γίνει κατανοητό και να εκτελεστεί το πρόγραμμα από τον υπολογιστή η γλώσσα Python διαθέτει ένα μεταφραστικό πρόγραμμα με το οποίο, μεταφράζεται σε **γλώσσα μηχανής**. Ο μεταφραστής:

1. Ελέγχει για συντακτικά λάθη το πρόγραμμα
2. Αν το πρόγραμμα έχει συντακτικά λάθη σταματάει η μετάφραση
3. Αν το πρόγραμμα δεν έχει συντακτικά λάθη το μετατρέπει σε γλώσσα μηχανής ώστε να μπορεί να εκτελεστεί από τον υπολογιστή



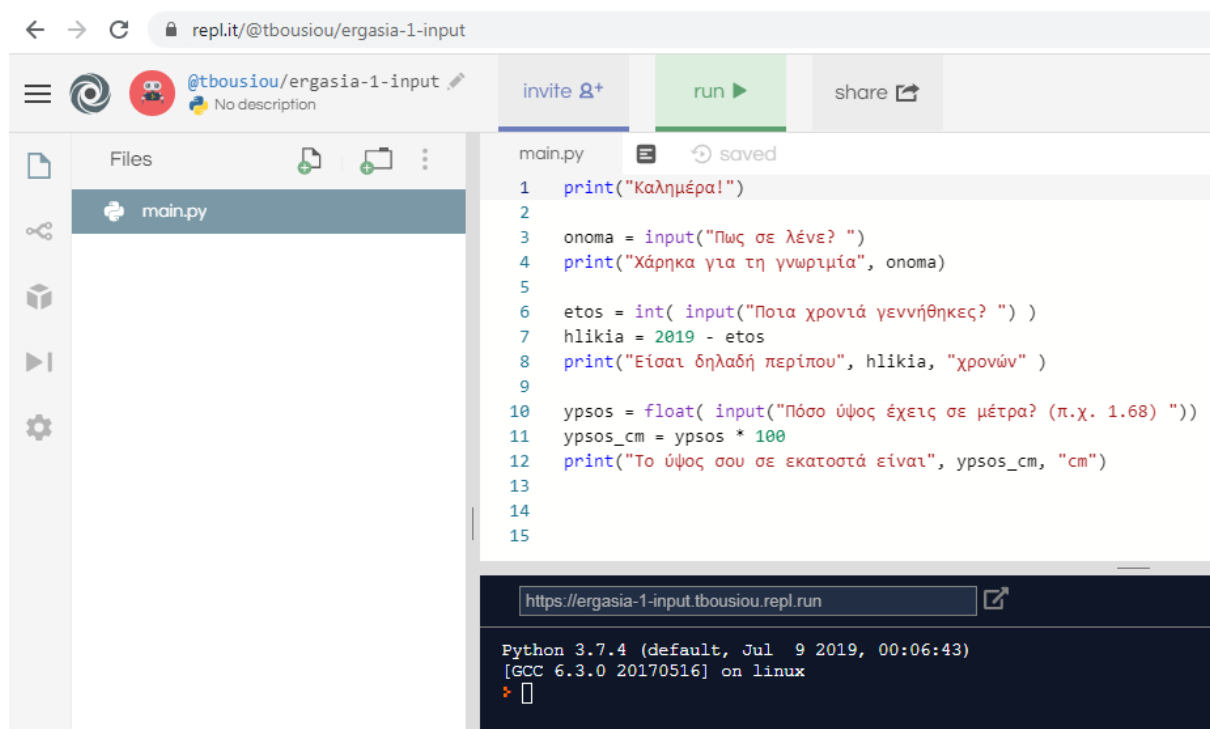
Περιβάλλον προγραμματισμού

Για την ανάπτυξη προγραμμάτων σε μια γλώσσα προγραμματισμού απαιτείται ένα σύνολο εργαλείων τα οποία ονομάζονται Ολοκληρωμένο Προγραμματιστικό Περιβάλλον (IDE Integrated Development Environment). Τα βασικά εργαλεία ενός IDE είναι:

- Επεξεργαστής κώδικα. Επεξεργαστής κειμένου με ειδικές λειτουργίες για τη συγγραφή κώδικα με γλώσσα προγραμματισμού.
- Μεταφραστής. Ελέγχει το πρόγραμμα για συντακτικά λάθη και το μετατρέπει σε εκτελέσιμο.
- Διάφορα βοηθητικά εργαλεία (διαχείριση αρχείων, εκσφαλμάτωση)

Περιβάλλον repl.it

Η εφαρμογή ιστού repl.it είναι ένα IDE που υποστηρίζει πολλές γλώσσες προγραμματισμού. Τα βασικά του πλεονέκτημα είναι η ευκολία στη χρήση και η δυνατότητα εκτέλεσης χωρίς εγκατάσταση μέσω browser. Για την ανάπτυξη σε γλώσσα Python μπορείτε να μεταβείτε σε αυτό το σύνδεσμο <https://repl.it/languages/python3>



Βασικά στοιχεία python

Μεταβλητές

Μεταβλητές είναι **θέσεις μνήμης** στις οποίες αποθηκεύονται τιμές. Στην Python δεν είναι απαραίτητο να δηλώσουμε τον τύπο της μεταβλητής που θα χρησιμοποιήσουμε, αλλά αυτό γίνεται αυτόματα, όταν η μεταβλητή πάρει τιμή την πρώτη φορά. Κάθε μεταβλητή έχει ένα όνομα και μια τιμή. Η τιμή μεταβλητής μπορεί να αλλάζει κατά την εκτέλεση ενός προγράμματος.

Εντολή εκχώρησης τιμής

Με την **εντολή εκχώρησης** τιμής δίνουμε τιμή σε μια μεταβλητή. Στο αριστερό μέρος πάντα μπαίνει το όνομα της μεταβλητής και δεξιά μια **έκφραση ή παράσταση**

```
ypsos = 5
platos = 2
perimetros = 2 * (platos + ypsos)
onoma = "Petros"
pi = 3.141345
ypsos = ypsos + 1
```

⚠ Τα ονόματα μεταβλητών περιέχουν μόνο λατινικά γράμματα, αριθμούς και τον χαρακτήρα κάτω παύλα (_). Το όνομα μεταβλητής αρχίζει πάντα από γράμμα.

Όνομα μεταβλητής	Σωστό ή Λάθος
ypsos	Σωστό
ypsos1	Σωστό, επιτρέπονται αριθμοί
ypsos_trigonou	Σωστό, ο χαρακτήρας _ επιτρέπεται
onoma@	Λάθος, δεν επιτρέπονται σύμβολα
1ypsos	Λάθος, πρέπει να αρχίζει από γράμμα

Τύποι δεδομένων

Ανάλογα με την τιμή μιας μεταβλητής καθορίζεται αυτόματα από την Python ο τύπος δεδομένων. Η Python υποστηρίζει τους παρακάτω τύπους δεδομένων

Τύπος	Παράδειγμα
Ακέραιος (integer)	5 12 131 0 -32 -100
Δεκαδικός (float)	3.234 0.3445 0.0 -12.41
Συμβολοσειρά (string)	"Maria" "Καλημέρα κόσμε!" "Α" "---"
Λογικός (boolean)	True False

⚠ Η υποδιαστολή στη γλώσσα Python είναι με **τελεία (.)** και όχι το κόμμα (,)

⚠ Οι συμβολοσειρές (κείμενο) πάντα περικλείονται από τον χαρακτήρα “

Αριθμητικές παραστάσεις

Μια αριθμητική παράσταση περιέχει σταθερές τιμές, μεταβλητές και τους αριθμητικούς τελεστές.

Πράξη	Τελεστής	Παράδειγμα
Πρόσθεση	+	$5 + 2 = 7$
Αφαίρεση	-	$5 - 2 = 3$
Πολλαπλασιασμός	*	$5 * 2 = 10$
Διαίρεση	/	$5 / 2 = 2.5$
Έκθεση σε δύναμη	**	$5 ** 2 = 25$
Υπόλοιπο διαίρεσης	%	$5 \% 2 = 1$

Παραδείγματα:

Έκφραση	Αποτέλεσμα a
$a = 5$	5
$a = 5 + 2 - 3$	4
$a = 5 * 2 + 10 / 2$	$10 + 5 = 15$
$a = 5 * (2 + 10 / 2)$	$5 * (2 + 5) = 5 * 7 = 35$
$a = 5 + 2 ** 3$	$5 + 2^3 = 5 + 8 = 13$
$a = (1 + 2) ** 3$	$3^3 = 27$

⚠ Στις αριθμητικές εκφράσεις ισχύει η **προτεραιότητα** των πράξεων (ύψωση σε δύναμη, πολλαπλασιασμός, διαίρεση, πρόσθεση και αφαίρεση)

Εντολή print()

Με τον εντολή `print()` εμφανίζουμε (ή εκτυπώνουμε) στην οθόνη του υπολογιστή ένα ή περισσότερα αποτελέσματα.

Σύνταξη:

`print(a)` Όπου `a` μπορεί να είναι σταθερά, μεταβλητή ή μια έκφραση γενικά.

`print(a, b, ...)` Πολλαπλά στοιχεία, τα διαχωρίζουμε με κόμμα

Παραδείγματα:

Εντολή	Αποτέλεσμα στην οθόνη
<code>print(5)</code>	5
<code>print(5+2)</code>	7
<code>a = 5</code> <code>print(2*a+1)</code>	11
<code>print("Καλημέρα")</code>	Καλημέρα
<code>print("Καλημέρα", "Γιάννη")</code>	Καλημέρα Γιάννη
<code>print(4, 4+2, 9)</code>	4 6 9
<code>name = "John"</code> <code>print("Hello", name)</code>	Hello John
<code>balance = 10</code> <code>print("Υπόλοιπο", balance, "Ευρώ")</code>	Υπόλοιπο 10 Ευρώ

Εντολή input()

Με την εντολή **`input()`** εισάγουμε δεδομένα από το πληκτρολόγιο στο πρόγραμμά μας.

Σύνταξη

`input("Μήνυμα")` Όπου μήνυμα μια συμβολοσειρά (κείμενο)

Εκτέλεση

Η εντολή `input()` όταν εκτελεστεί

1. Εμφανίζει ένα μήνυμα που ορίζουμε στην εντολή

2. Σταματά προσωρινά την εκτέλεση του προγράμματος
3. Ο χρήστης εισάγει δεδομένα με το πληκτρολόγιο
4. Όταν πατήσει το πλήκτρο Enter το πρόγραμμα συνεχίζει

α) Εισαγωγή δεδομένων κειμένου

Ότι εισάγουμε με την εντολή `input()` έχει τη μορφή **συμβολοσειράς** (κειμένου). Για να αποθηκεύσουμε αυτό που εισάγει ο χρήστης πρέπει να συνδυαστεί η εντολή `input` με εκχώρηση τιμής σε μεταβλητή

Εντολή	Χρήση
<code>onoma = input("Δώσε όνομα :")</code>	Εισάγουμε κείμενο και το αποθηκεύουμε σε μεταβλητή.

β) Εισαγωγή δεδομένων και μετατροπή σε αριθμό

Όταν πρέπει να εισάγουμε αριθμούς με την εντολή `input()` τότε πρέπει να τους μετατρέψουμε σε ακέραιο ή δεκαδικό με τις συναρτήσεις **`int()`** και **`float()`** αντίστοιχα, πριν την εκχώρηση σε μεταβλητή.

Εντολή	Εξήγηση
<code>age=int(input("Δώσε ηλικία σε χρόνια: "))</code>	Εισάγουμε αριθμό και τον μετατρέπουμε σε ακέραιο με την εντολή <code>int()</code>
<code>ypsos=float(input("Δώσε ύψος: "))</code>	Εισάγουμε αριθμό και τον μετατρέπουμε σε δεκαδικό με την εντολή <code>float()</code>

⚠ Η εντολή `input()` στις παραπάνω περιπτώσεις συνδυάζεται πάντοτε με εντολή εκχώρησης τιμής σε μεταβλητή.

γ) Προσωρινή παύση του προγράμματος

Μια απλή χρήση της εντολής `input()` είναι όταν θέλουμε να σταματήσουμε προσωρινά το πρόγραμμα και να συνεχίσουμε όταν ο χρήστης πατήσει το Enter. Σε αυτή την περίπτωση δε χρειάζεται εκχώρηση τιμής σε μεταβλητή.

Εντολή	Χρήση
<code>input("Πάτησε ENTER για συνέχεια...")</code>	Σταματάει το πρόγραμμα και περιμένει από το χρήστη να πατήσει ENTER

Τιμή μεταβλητών κατά την εκτέλεση

Μια μεταβλητή πάντα έχει **μια τιμή σε μια δεδομένη χρονική στιγμή**. Η τιμή αυτή όμως μπορεί να αλλάζει σε κάθε στάδιο της εκτέλεσης ενός προγράμματος, για παράδειγμα αν έχουμε τις παρακάτω εντολές:

```
a = 5
b = 2
a = b + a
```

τότε στον παρακάτω πίνακα φαίνονται οι τιμές των μεταβλητών σε κάθε βήμα (γραμμή) εκτέλεσης του προγράμματος

Βήμα / Τιμή	a	b
1	5	-
2	5	2
3	7	2

Προγράμματα με δομή ακολουθίας

Στα προγράμματα με δομή ακολουθίας οι εντολές εκτελούνται πάντοτε με την ίδια σειρά (ακολουθιακά). Σε ένα τέτοιο πρόγραμμα τα στάδια εκτέλεσης είναι συνήθως:

1. Είσοδος δεδομένων (με εντολές input)
2. Επεξεργασία ή Υπολογισμός (με εντολές εκχώρησης τιμής)
3. Εμφάνιση αποτελεσμάτων (με εντολές print)

Παράδειγμα 1

Να δημιουργήσετε πρόγραμμα σε γλώσσα Python που να ζητάει από τον χρήστη τη πλευρά ενός τετραγώνου, να υπολογίζει και να εμφανίζει το εμβαδόν του

```
plevra = float(input("Δώσε πλευρά σε cm "))
emvadon = plevra ** 2
print(emvadon)
```

Παράδειγμα 2

Να δημιουργήσετε πρόγραμμα σε γλώσσα Python που να ζητάει από τον χρήστη τη ποσότητα σε λίτρα βενζίνης και να υπολογίζει το κόστος, θεωρούμε ότι το λίτρο κοστίζει 1.78 Ευρώ

```
litra = float(input("Δώσε λίτρα βενζίνης "))
kostos = litra * 1.78
print("Κόστος βενζίνης:", kostos)
```

Παράδειγμα 3

Να δημιουργήσετε πρόγραμμα σε γλώσσα Python που να ζητάει από τον χρήστη το όνομά του και το έτος γέννησης. Στη συνέχεια το πρόγραμμα θα χαιρετά το χρήστη με το όνομά του και θα υπολογίζει και θα εμφανίζει την ηλικία του σε έτη

```
onoma = input("Πως σε λένε? ")
print("Γειά σου", onoma)
etos = int(input("Ποιό έτος γεννήθηκες? "))
ilikia = 2020 - etos
print("Πρέπει να είσαι τότε", ilikia, "χρονών")
```

Λογικές εκφράσεις και μεταβλητές

Μια λογική έκφραση πάντα υπολογίζεται σε τιμή **True** (Αληθής) ή **False** (Ψευδής) και σχηματίζεται με τη χρήση των **τελεστών σύγκρισης** και των **λογικών τελεστών**.

Παράδειγμα

`a > 1`

`a == 2 and b < 3`

Τελεστές σύγκρισης

Οι τελεστές σύγκρισης συγκρίνουν τιμές

Τελεστής	Εξήγηση
>	Μεγαλύτερο
>=	Μεγαλύτερο ή ίσο
<	Μικρότερο

<=	Μικρότερο ή ίσο
==	Ισότητα
!=	Διαφορετικό

Παραδείγματα:

Λογική έκφραση	Αποτέλεσμα
5 > 1	True
5 - 2 < 1	False
5 >= 2	True
2 <= 0	False
2 + 3 == 5	True
5 != 5	False
"John" == "Mary"	False
"John" != "Mary"	True
"α" > "β"	False
"cat" < "dog"	True

Λογικοί τελεστές

Τελεστής	Εξήγηση
and	Λογικό ΚΑΙ
or	Λογικό Η
not	Λογική Άρνηση

Παραδείγματα:

Λογική έκφραση	Αποτέλεσμα
5 > 1 and 5 < 8	True
5 > 1 and 5 < 4	False

<code>5 > 1 or 5 < 4</code>	True
<code>5 > 8 or 5 < 0</code>	False
<code>not 5 > 2</code>	False
<code>a=1 not a != 1</code>	True
<code>a = 5 (a > 1 and a < 0) or a == 5</code>	True

Λογικές μεταβλητές

Η τιμή μια λογικής έκφρασης μπορεί να αποθηκευτεί σε μεταβλητή. Μια λογική έκφραση υπολογίζεται πάντα στη τιμή **True** ή **False**

Παραδείγματα

```
done = True
game_over = lives < 0 or time_left < 0
isHot = thermokrasia >= 30
```

Δομή επιλογής

Με τη δομή επιλογής ένα πρόγραμμα παίρνει αποφάσεις ανάλογα με κάποια συνθήκη. Με αυτό το πρώτο η εκτέλεση των εντολών δεν είναι πάντα ακολουθιακή αλλά ακολουθούνται διαφορετικά μονοπάτια, ανάλογα την κατάσταση του προγράμματος. Η δομή επιλογής υλοποιείται με την εντολή `if` η οποία έχει τις παρακάτω μορφές.

Εντολή `if`

Με την απλή εντολή `if` εκτελείται ένα σύνολο (μπλοκ) εντολών ανάλογα με την τιμή μιας λογική έκφρασης ή συνθήκης. Αν συνθήκη είναι `True` τότε εκτελείται το μπλοκ εντολών, αλλιώς συνεχίζει το πρόγραμμα.

`if <συνθήκη>:`

μπλοκ
εντολών { εντολές αν η συνθήκη είναι
Αληθής

Παράδειγμα 1

```
password = input("Δώσε τον κωδικό ")
if (password == "secret"):
    print("Επιτυχής σύνδεση")
```

Παράδειγμα 2

```
age = int(input("Πόσο χρονών είσαι? "))
if (age < 18):
    print("Είσαι ανήλικος-η")
```

Εντολή if ... else ...

Η λειτουργία την εντολής **if ... else ...** είναι παρόμοια με την απλή **if**, με τη διαφορά ότι αν δεν ισχύει η συνθήκη τότε εκτελείται το μπλοκ εντολών μετά το **else**. Στην εντολή αυτή δηλαδή θα εκτελεστεί αναγκαστικά ένα από τα δύο μπλοκ.

if <συνθήκη>:

1.μπλοκ
εντολών { εντολές αν η συνθήκη είναι Αληθής

else:

2.μπλοκ
εντολών { εντολές αν η συνθήκη είναι Ψευδής

Παράδειγμα 1

```
age = int(input("Πόσο χρονών είσαι?"))
if (age < 18):
    print("Είσαι ανήλικος-η")
else:
    print("Είσαι ενήλικας")
```

Παράδειγμα 2

```
password = input("Δώσε τον κωδικό")
if (password == "secret"):
    print("Επιτυχή σύνδεση")
else:
    print("Λάθος κωδικός")
```

Εντολή if ... elif ... else ...

Στη μορφή αυτή έχουμε πολλαπλές συνθήκες οι οποίες ελέγχονται διαδοχικά. Εκτελείται το πρώτο μπλοκ εντολών που θα έχει την τιμή **True** στη συνθήκη. Η εντολή αυτή είναι χρήσιμη όταν έχουν πολλαπλές περιπτώσεις που πρέπει να ακολουθήσει ένας αλγόριθμος.

if <συνθήκη1>:

1.μπλοκ
εντολών { εντολές αν η συνθήκη1 είναι Αληθής

elif <συνθήκη2>:

2.μπλοκ
εντολών { εντολές αν η συνθήκη1 είναι Αληθής

else:

3.μπλοκ
εντολών { εντολές αν η συνθήκη1 και η
συνθήκη2 είναι Ψευδής
εντολές

Παράδειγμα

```
a = int(input("Δώσε ένα ακέραιο αριθμό"))
if (a > 0):
    print("Θετικός")
elif (a < 0):
    print("Αρνητικός")
else:
    print("Μηδέν")
```

Ασκήσεις με δομή επιλογής

Άσκηση 1

Να γραφεί πρόγραμμα σε γλώσσα Python που να επιλύει την εξίσωση $Ax+B=0$, για A και B πραγματικούς αριθμούς.

Λύση:

```
# Αν  $a \neq 0$ , τότε η εξίσωση  $ax + b = 0$  έχει μοναδική λύση την  $x = -b/a$   
# Αν  $a = 0$ , τότε η εξίσωση  $ax + b = 0$  γίνεται  $0x = -b$  και  
# αν  $b \neq 0$ , δεν έχει λύση (αδύνατη), ενώ  
# αν  $b = 0$ , κάθε αριθμός είναι λύση της
```

```
a = float(input("Δώσε το α"))  
b = float(input("Δώσε το β"))
```

```
if (a != 0):  
    x = -b/a  
    print("Μοναδική λύση x= ", x)  
else:  
    if (b != 0):  
        print("Δεν έχει λύση, αδύνατη")  
    else:  
        print("Κάθε αριθμός είναι λύση")
```

Άσκηση 2

Σε ένα ηλεκτρικό κύκλωμα υπάρχουν δύο αντιστάσεις R_1 και R_2 . Θέλουμε να βρούμε την ολική αντίσταση R του κυκλώματος. Υπάρχουν δύο τρόποι σύνδεσης αυτών των αντιστάσεων: σε σειρά ή παράλληλα. Στην πρώτη περίπτωση, ο τύπος που δίνει την ολική αντίσταση είναι $R = R_1 + R_2$. Στην παράλληλη σύνδεση, η ολική αντίσταση βρίσκεται από τον τύπο $R = (R_1 * R_2) / (R_1 + R_2)$. Να γραφεί πρόγραμμα σε γλώσσα Python που να:

- διαβάζει τις τιμές των αντιστάσεων R_1 και R_2
- διαβάζει τον τρόπο σύνδεσης, δίνοντας την επιλογή 1 για σύνδεση σε σειρά ή τη 2 για παράλληλη σύνδεση, των αντιστάσεων R_1 και R_2 .
- υπολογίζει, εφαρμόζοντας τον κατάλληλο τύπο ανάλογα με τον τρόπο σύνδεσης, την τιμή της ολικής αντίστασης R .
- εμφανίζει τις τιμές των αντιστάσεων R_1 , R_2 , καθώς και την τιμή της αντίστασης R .

Λύση

```

# Διάβασε τιμές αντιστάσεων
r1 = float(input("Δώσε τιμή R1 "))
r2 = float(input("Δώσε τιμή R2 "))

# Διάβασε τρόπο σύνδεσης
choice = int(input("Δώσε τρόπο σύνδεσης, 1 σε σειρά, 2 παράλληλα"))

# Υπολόγισε ολική αντίσταση και εμφάνισε
if (choice == 1):
    r = r1 + r2
    print("Ολική αντίσταση σε σειρά Rs= ", r)
elif (choice == 2):
    r = (r1*r2) / (r1+r2)
    print("Ολική αντίσταση σε παράλληλη σύνδεση Rp= ", r)
else:
    print("Λάθος επιλογή, δώσε 1 ή 2")

```

Συναρτήσεις

Οι συναρτήσεις είναι επαναχρησιμοποιήσιμα μέρη προγραμμάτων. Μας επιτρέπουν να δίνουμε ένα όνομα σε ένα σύνολο εντολών και να το εκτελούμε καλώντας το όνομά τους, οπουδήποτε στο πρόγραμμα και όσες φορές θέλουμε. Αυτή η διαδικασία ονομάζεται κλήση (call) της συνάρτησης.

Ορισμός και κλήση συνάρτησης

Μια συνάρτηση πρέπει πρώτα να οριστεί και μετά μπορεί να γίνει η κλήση της στο πρόγραμμα. Ο ορισμός γίνεται με την εντολή def.

```

#Ορισμός συνάρτησης
def sayHello():
    print("Hello")
    print("My name is James Bond")

#Κλήση συνάρτησης
sayHello()

```

Παράμετροι συνάρτησης

Μια συνάρτηση μπορεί να έχει μια ή περισσότερες παραμέτρους κατά τον ορισμό της. Κατά την κλήση της τότε μπορούν να περαστούν σταθερές τιμές ή μεταβλητές στη λίστα παραμέτρων

```
#Ορισμός συνάρτησης
def sayHello(name,surname):
    print("Hello")
    print("My name is",name, surname)

#Κλήση συνάρτησης
sayHello("James", "Bond")

myname = "John"
mysurname = "Doe"
sayHello(myname, mysurname)
```

Επιστροφή τιμής από συνάρτηση

Μια συνάρτηση μπορεί να επιστρέφει μια τιμή. Η επιστροφή της τιμής γίνεται με εντολή **return** με την οποία και τερματίζει η κλήση της συνάρτησης. Όταν μια συνάρτηση επιστρέφει τιμή μπορεί να υπάρχει και σε μια έκφραση.

```
#Ορισμός συνάρτησης
def mesos_oros(a, b):
    mo = (a + b) / 2
    return mo

#Κλήση συνάρτησης
n1 = 5
n2 = 7
a = mesos_oros(n1, n2)
b = mesos_oros(3, 9)
c = mesos_oros(3, 9) + 1

print(a, b, c)
```

Το αποτέλεσμα στην οθόνη θα είναι **6.0 6.0 7.0**

Παράδειγμα 1

Να ορίσετε συνάρτηση για τον υπολογισμό της διακρίνουσας $\Delta = b^2 - 4ac$ της δευτεροβάθμιας εξίσωσης

#Ορισμός συνάρτησης

```
def diakrinousa(b, c):  
    d = b**2 - 4 * b * c  
    return d
```

Παράδειγμα 2

Να φτιάξετε ένα πρόγραμμα που να ελέγχει τα στοιχεία σύνδεσης χρήστη (όνομα χρήστη και κωδικό) και να δίνει το κατάλληλο μήνυμα. Ο έλεγχος να γίνεται με χρήση συνάρτησης που να επιστρέφει True ή False. Θεωρούμε ότι τα σωστά στοιχεία είναι "admin" και "secret".

```
def check_login(username, password):  
    """ username το όνομα χρήστη  
        password ο κωδικός  
    """  
    if username == "admin" and password == "secret":  
        return True  
    else:  
        return False
```

```
u = input("Όνομα χρήστη: ")  
p = input("Κωδικός: ")  
  
if check_login(u, p):  
    print("Συνδεθήκατε με επιτυχία")  
else:  
    print("Λάθος στοιχεία")
```

Σχόλια σε πρόγραμμα

Τα **σχόλια (comments)** είναι σημεία γραμμές ή μέρη του κώδικα τα οποία αγνοεί ο μεταφραστής, δεν έχουν δηλαδή καμία επιρροή στο πρόγραμμα. Είναι όμως πολύ σημαντικά γιατί

- Περιγράφουν τη λειτουργία του κώδικα. Με αυτό τον τρόπο διευκολύνουμε την ανάγνωση του κώδικα από άλλους αλλά και για εμάς τους ίδιους.
- Κατά την εκσφαλμάτωση μπορούμε με τα σχόλια να απενεργοποιήσουμε προσωρινά τμήματα κώδικα.

Τα σχόλια γραμμής στην python γίνονται με τον χαρακτήρα **#**

Τα σχόλια πολλαπλών γραμμών περικλείονται με τριπλά εισαγωγικά **"""**

```
def check_login(username, password):  
    """ username το όνομα χρήστη  
    | password ο κωδικός  
    | """  
    if username == "admin" and password == "secret":  
        return True  
    else:  
        return False  
  
# Διάβασε δεδομένα  
u = input("Όνομα χρήστη: ") # Διάβασε όνομα χρήστη  
p = input("Κωδικός: ")
```

Ενσωματωμένες συναρτήσεις

abs()	Επιστρέφει την απόλυτη τιμή αριθμού	abs(-3) -> 3
bool()	Μετατρέπει σε λογική τιμή	bool(5) -> True bool(0) -> False
chr()	Μετατρέπει την ascii τιμή στον αντίστοιχο χαρακτήρα	chr(97) -> "a"
ord()	Μετατρέπει έναν χαρακτήρα στη τιμή ascii	ord("a") -> 97
len()	Επιστρέφει το μήκος	len("Hello") -> 5

	συμβολοσειράς ή λίστας	
max(), min()	Μέγιστο ή ελάχιστο σειράς ή λίστας αριθμών	max(4,7,2) -> 7
sum()	Επιστρέφει το άθροισμα λίστας αριθμών	sum(5, 3, 2) -> 10

Modules

Η python διαθέτει μια τεράστια **βιβλιοθήκη** με συναρτήσεις που βρίσκονται στα αρχεία εγκατάστασης της Python. Τα αρχεία αυτά ονομάζονται **modules**. Για να χρησιμοποιήσουμε μια συνάρτηση από ένα module της Python πρέπει να το εισάγουμε στο κώδικά με την εντολή **import**

```
# A) Εισαγωγή όλου του module
import math

print (math.sqrt(2)) # Τετραγωνική ρίζα
print (math.sin(1))  # Ημίτονο

# B) Εισαγωγή συγκεκριμένων συναρτήσεων
from math import sqrt, sin

print(sqrt(2))
print(sin(1))
```

Χρήσιμες συναρτήσεις από άλλα modules

random.randint(a, b)	Επιστρέφει έναν τυχαίο ακέραιο αριθμό μεταξύ a και b	randint(4, 11) -> 9
random.choice(list)	Επιστρέφει έναν τυχαίο αριθμό από τη λίστα	choice[4,3,1,9] -> 3
time.sleep(secs)	Σταματά το πρόγραμμα για τα δευτερόλεπτα που ορίζονται ως παράμετρο	
os.system(command)	Εκτελεί μια εντολή του λειτουργικού συστήματος	system("clear")