Δομή Ακολουθίας και Μεταβλητές

Οι εντολές εκτελούνται με την σειρά που γράφονται. Η τελευταία εντολή σηματοδοτεί το τέλος του προγράμματος.

```
1  name = "John"
2  print(name)
3
4  firstname, lastname = "John", "Giftakis"
5  print(f"Είμαι ο {firstname} {lastname}")
```

Παράδειγμα 1-1. Θέλουμε να κατεβάσουμε στο κινητό μας τηλέφωνο μια εφαρμογή. Η ταχύτητα σύνδεσής μας στο διαδίκτυο είναι 1,6 MB/δευτερόλεπτο. Γράψτε σε γλώσσα Python πρόγραμμα, που δέχεται ως είσοδο το μέγεθος της εφαρμογής σε MB και υπολογίζει σε πόσο χρόνο θα κατέβει η εφαρμογή.

Αλγόριθμος Download	#Download
Γράψε 'Δώσε το μέγεθος της εφαρμογής'	megethos=input("Δώσε το μέγεθος της εγγραφής")
Διάβασε μέγεθος	time=megethos/1.6
Χρόνος<-- μέγεθος/1,6	print time
Γράψε χρόνος	
Τέλος Download	

Παράδειγμα 1-2: Να δημιουργήσετε πρόγραμμα σε γλώσσα Python που να δίνετε από το πληκτρολόγιο το ημερομίσθιο ενός εργαζομένου και στην συνέχεια να υπολογίζετε και το εμφανίζετε το ποσό που θα εισπράξει μετά από 25 ημέρες εργασίας.

Αλγόριθμος Ημερομίσθιο	#Hmeromisthio
Γράψε 'Δώσε το ημερομίσθιο'	imeromisthio=input("Δώσε το ημερομίσθιο:")
Διάβασε ημερομίσθιο	misthos= imeromisthio*25
Μισθός<-- ημερομίσθιο*25	print "Μισθός=",misthos
Γράψε 'Μισθός=',Μισθός	
Τέλος Ημερομίσθιο	

Παράδειγμα 1-3: Να δημιουργήσετε πρόγραμμα σε γλώσσα προγραμματισμού Python, όπου θα δίνετε την ακτίνα του κύκλου και να υπολογίζετε και να εμφανίζετε το μήκος της περιφέρειας και το εμβαδόν του κύκλου.

Αλγόριθμος Κύκλος	#periferia-emvadon circle
Γράψε 'Δώσε ακτίνα κύκλου'	a=float(input('Δώσε ακτίνα κύκλου'))
Διάβασε ακτίνα	pi=3.14
Π<--3.14	periferia=a*2*pi
Περιφέρεια<-- ακτίνα*2*π	emvadon=a**2*pi
Εμβαδόν<-- ακτίνα^2*π	print 'Μήκος περιφέρειας κύκλου:',periferia
Γράψε 'Περιφέρεια κύκλου=',Περιφέρεια	print 'Εμβαδόν κύκλο:', emvadon
Γράψε 'Εμβαδόν κύκλου=', Εμβαδόν	
Τέλος Κύκλος	

Παρατήρηση

Η μεταβλητή a (=ακτίνα κύκλου) είναι πραγματική. Η Python δεν απαιτεί να δηλώσουμε τον τύπο της μεταβλητής a πριν τη χρησιμοποιήσουμε, αλλά για να διασφαλίσουμε ότι η τιμή που θα δοθεί είναι πραγματικός αριθμός, μετατρέπουμε την είσοδο σε πραγματικό αριθμό με τη συνάρτηση float(). Παρόμοια, μια τιμή μπορεί να μετατραπεί σε ακέραια με τη συνάρτηση int(): π.χ. όταν θέλουμε να διαβάσουμε ακέραιο αριθμό από το πληκτρολόγιο χρησιμοποιούμε την εντολή: a=int(input('Δώσε ακέραιο αριθμό:'))

```
a = 15
b = 2
print(a + b) #πρόσθεση
print(a - b) #αφαίρεση
print(a * b) #πολλαπλασιασμός
print(a / b) #διαίρεση
print(a // b) #ακέραια διαίρεση
print(a % b) #ακέραιο υπόλοιπο
print(a ** b) #ύψωση σε δύναμη
```

```
# Δημιουργία μιας μεταβλητής με τιμή ακέραιου αριθμού
x = 5
print(x)

# Δημιουργία μιας μεταβλητής με τιμή κειμένου
name = "John"
print(name)

# Δημιουργία μιας μεταβλητής με τιμή αληθούς (Boolean)
is_cool = True
print(is_cool)
```

Μία μεταβλητή μπορεί να περιέχει μία πληροφορία κάθε φορά.

```
>>> name="Marios"
>>> name="Eleni"
>>> name="Kostas"
>>> print(name)
Kostas
```

```
>>> name="Alexis"
>>> surname="Ioannou"
>>> print(name,surname)
Alexis Ioannou
```

Προσέξτε το κόμα (,)

Με παρόμοιο τρόπο μπορούμε να δώσουμε αριθμητική τιμή σε μια μεταβλητή.

```
>>> arithmos=10
```

Στο πιο πάνω παράδειγμα, δημιουργήσαμε μια μεταβλητή με το όνομα arithmos και της δώσαμε την τιμή 10. Το 10 είναι αριθμός, έτσι δεν χρειάζεται εισαγωγικά. Ας δούμε τώρα πώς να χρησιμοποιήσουμε μεταβλητές σε μαθηματικές πράξεις:

```
>>> print(arithmos+5)
15
```

10+5 = 15

Με την εντολή print, ζητήσαμε να προσθέσει το περιεχόμενο της μεταβλητής arithmos (που είναι το 10) με τον αριθμό 5. Έτσι, το αποτέλεσμα είναι 15.

```
>>> mytext="Δε θα ξαναμιλήσω στο μάθημα"
```

Στη μεταβλητή mytext, έχουμε τοποθετήσει τη φράση "Δε θα ξαναμιλήσω στο μάθημα". Τώρα θα δούμε πώς μπορούμε να την επαναλάβουμε με μια απλή εντολή:

```
>>> print(mytext*100)
```

Η εντολή πιο πάνω εμφανίζει το περιεχόμενο της μεταβλητής mytext, ενώ το σύμβολο * και ο αριθμός 100 επαναλαμβάνουν την εμφάνιση του στην οθόνη.

```
>>> print(keimeno*100)
δε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στ
ο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμ
ιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε
θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μ
άθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλή
σω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα
ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθη
μαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω στο μάθημαδε θα ξαναμιλήσω
```

```
>>> firstnumber=10
```

Έχουμε δημιουργήσει μια μεταβλητή με το όνομα firstnumber και της έχουμε δώσει την τιμή 10

```
>>> secondnumber=firstnumber
```

Έχουμε δημιουργήσει μια μεταβλητή με το όνομα secondnumber και της έχουμε δώσει την τιμή που περιέχει η μεταβλητή firstnumber.

```
>>> print(secondnumber)
    10
```

```
>>> programlang="Python"
>>> print(programlang*4)
    PythonPythonPythonPython
```

Python * 4 φορές

Τι συμβαίνει όταν προσθέσουμε έναν ακέραιο με μια μεταβλητή που περιέχει κείμενο;

```
>>> print(programlang+5)
```

Θα βγάλει Error

Όταν πατήσουμε το ENTER για να εκτελεστεί η πιο πάνω εντολή, παίρνουμε το ακόλουθο μήνυμα:

```
Traceback (most recent call last):
  File "<pyshell#18>", line 1, in
  <module>
    print(programlang+5)

TypeError: can only concatenate str
(not "int") to str
```

Δεν μπορούμε να προσθέσουμε **ακέραιο (int)** σε μεταβλητή που περιέχει **κείμενο (str)**!

```
>>> programlang1="My favorite language is"
>>> programlang2="Python"
```

Στις πιο πάνω εντολές έχουμε δημιουργήσει δύο μεταβλητές. Ας δούμε τώρα πώς εμφανίζονται με την εκτέλεση:

με +

```
>>> print(programlang1+programlang2)
    My favorite language isPython
```

Το σύμβολο "+" έχει ενώσει το περιεχόμενο των δύο μεταβλητών. Δεν έχει, όμως, αφήσει απόσταση ανάμεσα στο κείμενο της μίας μεταβλητής και στο κείμενο της άλλης μεταβλητής! Σε μια τέτοια περίπτωση, αν επιθυμούμε να υπάρχει απόσταση, χρησιμοποιούμε το "," αντί για το "+".

με ,

```
>>> print(programlang1,programlang2)
    My favorite language is Python
```

Στον προγραμματισμό, το casting είναι η διαδικασία μετατροπής μιας μεταβλητής από έναν τύπο δεδομένων σε έναν άλλο τύπο δεδομένων. Το casting είναι χρήσιμο όταν θέλουμε να εκτελέσουμε μια πράξη ή μια συνάρτηση που δέχεται μια διαφορετική κλάση αντικειμένου από τον τύπο της μεταβλητής μας. Για παράδειγμα, μπορούμε να μετατρέψουμε μια μεταβλητή από ακέραιο αριθμό σε δεκαδικό αριθμό για να εκτελέσουμε μια πράξη που απαιτεί δεκαδικό αριθμό.

Στην Python, το casting γίνεται χρησιμοποιώντας τους τελεστές int(), float(), str() ανάλογα με τον τύπο δεδομένων που θέλουμε να μετατρέψουμε τη μεταβλητή μας. Για παράδειγμα, αν θέλουμε να μετατρέψουμε μια μεταβλητή από ακέραιο σε δεκαδικό αριθμό, χρησιμοποιούμε τον τελεστή float() για να μετατρέψουμε τη μεταβλητή μας σε δεκαδικό αριθμό.

Παρακάτω παρουσιάζονται μερικά παραδείγματα του πώς λειτουργεί το casting στην Python:

Παρακάτω παρουσιάζονται μερικά παραδείγματα του πώς λειτουργεί το casting στην Python:

```
# Casting από int σε float
x = 5
y = float(x)
print(y) # Output: 5.0

# Casting από float σε int
x = 2.7
y = int(x)
print(y) # Output: 2

# Casting από string σε int
x = "3"
y = int(x)
print(y) # Output: 3

# Casting από string σε float
x = "3.2"
y = float(x)
print(y) # Output: 3.2

# Casting από int σε string
x = 5
y = str(x)
print(y) # Output: "5"
```

Η έξοδος βγάζει χαρακτήρα = str(x)

Στο παραπάνω παράδειγμα, βλέπουμε πώς μπορούμε να χρησιμοποιήσουμε τη συνάρτηση float(), int() ή str() για να κάνουμε casting από έναν τύπο δεδομένων σε έναν άλλο τύπο δεδομένων. Παρατηρούμε ότι το αποτέλεσμα του casting είναι μια μεταβλητή με τον νέο τύπο δεδομένων που προσδιορίζουμε με τη συνάρτηση.

Για παράδειγμα, μπορείτε να ορίσετε μια συμβολοσειρά χρησιμοποιώντας μονά εισαγωγικά και να περιλάβετε διπλά εισαγωγικά μέσα της:

```
my_string = 'Αυτή είναι μια συμβολοσειρά που περιέχει "διπλά εισαγωγικά" μέσα της.'
```

Αντίστοιχα, μπορείτε να χρησιμοποιήσετε διπλά εισαγωγικά για τη δήλωση μιας συμβολοσειράς που περιέχει μονά εισαγωγικά:

```
my_string = "Αυτή είναι μια συμβολοσειρά που περιέχει 'μονά εισαγωγικά' μέσα της."
```

Ουσιαστικά, μπορείτε να επιλέξετε τα εισαγωγικά που είναι πιο βολικά για εσάς, ανάλογα με τις απαιτήσεις του κώδικά σας.

Αν η συμβολοσειρά περιέχει μόνα εισαγωγικά, τότε μπορεί να δηλωθεί με διπλά εισαγωγικά, και το αντίστροφο. Παραδείγματος χάρη:

```
x = "Αυτή η συμβολοσειρά περιέχει μονά 'εισαγωγικά'"
y = 'Αυτή η συμβολοσειρά περιέχει διπλά "εισαγωγικά"'
```

Ωστόσο, αν η συμβολοσειρά περιέχει τόσο μονά όσο και διπλά εισαγωγικά, τότε πρέπει να γίνει χρήση του αντίθετου τύπου εισαγωγικού για τη δήλωση της συμβολοσειράς, ή να χρησιμοποιηθούν αποδιορθωτικά στοιχεία (escape characters). Παραδείγματος χάρη:

```
x = "Αυτή η συμβολοσειρά περιέχει διπλά \"εισαγωγικά\" και μονά 'εισαγωγικά'"
y = 'Αυτή η συμβολοσειρά περιέχει μονά \'εισαγωγικά\' και διπλά "εισαγωγικά'
```

Στην Python, οι ονομασίες μεταβλητών (variable names) είναι case-sensitive, που σημαίνει ότι οι μεταβλητές x, X και x1 θεωρούνται διαφορετικές μεταξύ τους.

Παραδείγματα:

```
x = 5
X = "Αυτό είναι μια συμβολοσειρά"
print(x)  # Εκτύπωση της μεταβλητής x
print(X)  # Εκτύπωση της μεταβλητής X

x1 = "Αυτή είναι μια ακόμα συμβολοσειρά"
print(x1)  # Εκτύπωση της μεταβλητής x1

# Προσπάθεια εκτύπωσης μιας μεταβλητής με λάθος ονομασία
print(X1)  # Προκαλεί σφάλμα (NameError) καθώς η μεταβλητή X1 δεν έχει οριστεί
```

Στην συνάρτηση print(), όταν προσπαθείτε να συνδυάσετε μια συμβολοσειρά και έναν αριθμό με τον τελεστή +, η Python θα σας δώσει ένα σφάλμα. Πρέπει να μετατρέψετε τον αριθμό σε μια συμβολοσειρά χρησιμοποιώντας τη συνάρτηση str() πριν τον συνδυάσετε με μια συμβολοσειρά. Για παράδειγμα:

```
# Δημιουργία μεταβλητής x και εκχώρηση τιμής 5
x = 5

# Εκτύπωση του αριθμού x
print("Ο αριθμός είναι: " + str(x))
```

Στο παραπάνω παράδειγμα, η μεταβλητή x λαμβάνει την τιμή 5 και η συνάρτηση print() χρησιμοποιείται για να εκτυπώσει τη συμβολοσειρά "Ο αριθμός είναι: " και την τιμή της μεταβλητής x. Επειδή η μεταβλητή x είναι αριθμός, πρέπει να μετατραπεί σε συμβολοσειρά χρησιμοποιώντας τη συνάρτηση str(). Το αποτέλεσμα που θα εμφανιστεί θα είναι:

```
Ο αριθμός είναι: 5
```

Ο καλύτερος τρόπος για να εκτυπώσετε πολλαπλές μεταβλητές στη συνάρτηση print() είναι να τις χωρίσετε με κόμματα, που υποστηρίζουν ακόμα και διαφορετικούς τύπους δεδομένων:

```
# Δημιουργία μεταβλητής x με τιμή "John"
x = "John"

# Δημιουργία μεταβλητής y με τιμή 23
y = 23

# Εκτύπωση του μηνύματος με τις τιμές των μεταβλητών x και y
print("My name is", x, "and I am", y)
```

Στο παραπάνω παράδειγμα, οι μεταβλητές x και y παίρνουν τις τιμές "John" και 23 αντίστοιχα και η συνάρτηση print() χρησιμοποιείται για να εκτυπώσει τη συμβολοσειρά "My name is", την τιμή της μεταβλητής x, τη συμβολοσειρά "and I am" και την τιμή της μεταβλητής y. Η συνάρτηση print() θα εκτυπώσει την παρακάτω γραμμή:

```
My name is John and I am 23
```

Αναφορές

- Ευσταθίου Χρήστος, *Αρχές Προγραμματισμού Υπολογιστώ σε γλώσσα Python*, ΕΠΑΛ Τρικάλων, 2021
- Κοφτερός Αλέξανδρος, *Η Python με Απλά Λόγια: Σύντομος οδηγός για αρχάριους*, Λευκωσία, 2023
- Γυφτάκης Γιάννης, *https://courses.giftakis.gr/*, ανακτήθηκε 21/10/2023
- https://pliroforiki-edu.gr/, ανακτήθηκε 21/10/2023