ΠΕΡΙΕΧΟΜΕΝΑ:

Πέτρος Γ.

Χρυσός Χορηγός Μαθήματος

Αβραάμ

Χρυσός Χορηγός Μαθήματος

- Με χρήση απλών παραστάσεων (εκφράσεις, expressions), μπορούμε να κατασκευάσουμε άλλες, πιο συνθετες χρησιμοποιώντας **τελεστές (operators)**
  - Υπάρχουν οι αριθμητικοί, σχεσιακοί, λογικοί τελεστές καθώς και οι τελεστές καταχώρησης και bitwise τελεστές.

- **Αριθμητικοί Τελεστές** (arithmetic)
  - **+, -, *, /, //, %**
- (βλ. Μαθ. 2)

- **Τελεστές Καταχώρησης** (assignment)
  - **=**
  - **+=, -=, *=, /=**
- (βλ. Μαθ. 2)

- **Bitwise Τελεστές**
  - **&** (bitwise and (και))
  - **|** (bitwise or (ή))
  - **^** (bitwise xor (αποκλειστικό ή)
  - **~** (not)
  - **<<** (shift left – αριστερή ολίσθηση)
  - **>>** (shift right – δεξιά ολίσθηση)
- (θα καλυφθούν πολύ αργότερα ;-)

- **Σχεσιακοί Τελεστές**
  - **==** (ίσα)
  - **!=** (όχι ίσα)
  - **>, <, >=, <=** (μεγαλύτερο από κ.ο.κ.)
- Συγκρίνουν δύο αριθμούς ή δύο strings
- Επιστρέφουν True/False
- Ειδικά για τα strings θεωρείται ότι τα μικρά γράμματα είναι λεξικογραφικά μεγαλύτερα από τα κεφαλαία

Παράδειγμα 1: comparison.py

```
print(8+1 == 9)
print(6*2 != 7-3)
print(4%2 >= 5/4)
print(6-5 < 1%3)

print("abc" < "def")
print("abc" < "abcd")
print("abc" < "ABC")
```

- **Λογικοί Τελεστές**
  - and (λογικό και)
    - Επιστρέφει True αν και οι δύο εκφράσεις είναι True
  - or (λογικό ή)
    - Επιστρέφει True αν τουλάχιστον η μία έκφραση είναι True
  - not (λογικό όχι)
    - Αντιστρέφει την τιμή της έκφρασης
- Επενεργούν πάνω σε λογικές μεταβλητές.

Παράδειγμα 2: logical.py

```
spirit = True
flesh = False
print(spirit and flesh)
print(spirit or flesh)
print(not spirit)
print(spirit and not flesh)
```

Άσκηση 1: Προτεραιότητα τελεστών

- Οι σχεσιακοί τελεστές έχουν μικρότερη προτεραιότητα από τους αριθμητικούς τελεστές
  - Για τον λόγο αυτό γίνονται πρώτα οι αριθμητικές πράξεις και έπειτα η εφαρμογή των σχεσιακών τελεστών.
  - Γράψτε ένα πρόγραμμα το οποίο να ενσωματώνει σε μία παράσταση αριθμητικούς και έναν σχεσιακό τελεστή.
- Οι λογικοί τελεστές έχουν μικρότερη προτεραιότητα από τους σχεσιακούς τελεστές
  - Για τον λόγο αυτό οι λογικοί τελεστές εφαρμόζονται τελευταίοι
  - Γράψτε ένα πρόγραμμα το οποίο να ενσωματώνει σε μία παράσταση αριθμητικούς τελεστές, δύο σχεσιακούς τελεστές και έναν λογικό τελεστή.

Άσκηση 2: The geek factor

Κατασκευάστε ένα πρόγραμμα το οποίο:

- Θα ορίζει μία λογική μεταβλητή με όνομα is_geek
  - Αρχικοποιήστε την με την αυτοαξιολόγησή σας για το αν είστε geek
- Θα ορίζει μία ακέραια μεταβλητή με όνομα computer_hours
  - Αρχικοποιήστε την με την αυτοαξιολόγησή σας για το πόσες ώρες κάθεστε στον υπολογιστή ανά ημέρα.
- Θα ορίζει μία ακέραια μεταβλητή με όνομα sport_hours
  - Αρχικοποιήστε την με την αυτοαξιολόγησή σας για το πόσες ώρες αθλείστε ανά ημέρα.
- Αποθηκεύστε σε μία νέα λογική μεταβλητή geek_factor:
  - "Κάθομαι στον υπολογιστή περισσότερες από 5 ώρες, αθλούμαι λιγότερες από 2 ώρες και δεν πιστεύω ότι είμαι geek"
  - Ερμηνεύστε το αποτέλεσμα για τους ελέγχους σας.

- Σημείωση: Η προτεραιότητα των λογικών τελεστών είναι:
  - or (μικρότερη προτεραιότητα)
  - and
  - not

- Με τη δομή ελέγχου if μπορούμε να εκτελούμε διαφορετικές εντολές ανάλογα με το αν ισχύει μια συνθήκη.
  - Η συνθήκη συντάσσεται συνήθως με χρήση των λογικών και των σχεσιακών τελεστών που είδαμε προηγουμένως.
  - Έτσι, με την if μπορούμε να κάνουμε διαφορετικές ενέργειες ανάλογα με το αν ικανοποιούνται συνθήκες που εμείς ορίζουμε.

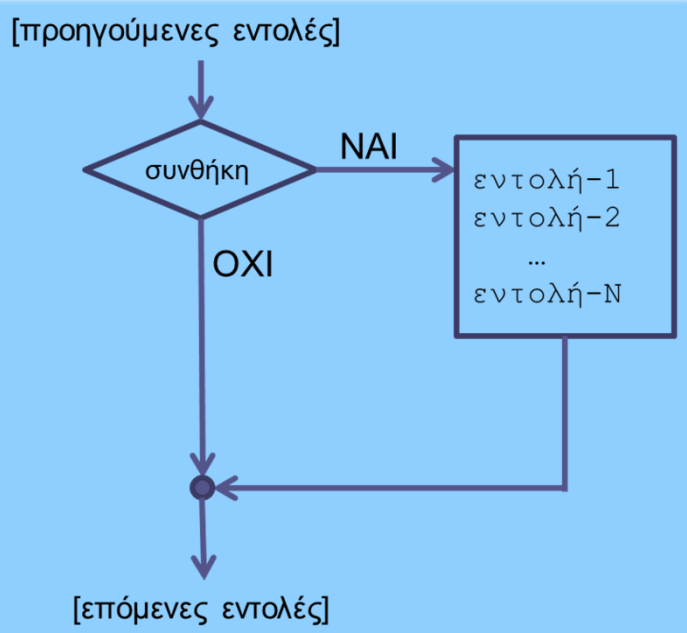
Απλή if

- Συντακτικό:

```
[προηγούμενες εντολές]

if συνθήκη:
   εντολή-1
   ...
   εντολή-N

[επόμενες εντολές]
```



- Εξήγηση:
  - Αφού εκτελεστούν οι προηγούμενες εντολές
  - Ελέγχεται η συνθήκη (συνήθως θα είναι μια παράσταση που θα χρησιμοποιεί κάποιον σχεσιακό/λογικό τελεστή)
    - Αν η συνθήκη είναι True εκτελούνται οι εντολές μέσα στο if και έπειτα οι (επόμενες εντολές)
    - Αν η συνθήκη είναι False εκτελούνται απευθείας οι (επόμενες εντολές)

Προσοχή:

- Μετα τη συνθήκη πρέπει να υπάρχει η άνω κάτω τελεία
- Οι εντολές μέσα στο if, πρέπει να ακολουθούν κανόνες στοίχισης (identation rules):
  - Πρέπει να βάζουμε πάντα ένα tab (ή ίσο αριθμό κενών) στις εντολές μιας if, αλλιώς προκύπτει συντακτικό λάθος.
- Ορολογία: Οι εντολές που είναι μέσα στην if λέμε ότι ορίζουν ένα **μπλοκ κώδικα.**

Παράδειγμα 3: comparison.py

```
sea_wave_height = 1.30
print("I am going to the beach")

if sea_wave_height <= 0.50:
   print("I will swim")

print("I will sunbath")
```

Άσκηση 3: Διαδοχικές if

- Γράψτε ένα πρόγραμμα το οποίο θα δέχεται από την είσοδο δύο ακεραίους και έπειτα θα ελέγχει αν ισχύει κάποιος σχεσιακός τελεστής μεταξύ δύο αριθμών.
- Π.χ. για το ίσον:

```
....
if x == y:
    print(str(x) + "==" + str(y))
...
```

- Το πρόγραμμα να ενσωματώνει τους σχεσιακούς τελεστές ==, !=, <, >, <=, >=

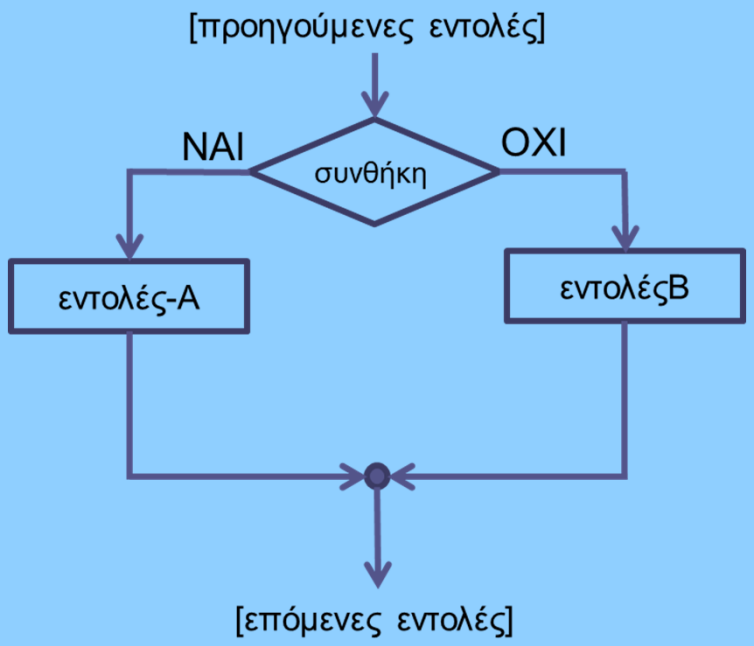
if...else

- Συντακτικό:

```
[προηγούμενες εντολές]

if συνθήκη:
    εντολές-Α
else:
    εντολές-Β

[επόμενες εντολές]
```



- Εξήγηση:
  - Αφού εκτελεστούν οι προηγούμενες εντολές
  - Ελέγχεται η συνθήκη (συνήθως θα είναι μια παράσταση που θα χρησιμοποιεί κάποιον σχεσιακό/λογικό τελεστή)
    - Αν η συνθήκη είναι True εκτελούνται οι εντολές μέσα στο if και έπειτα οι (επόμενες εντολές)
    - Αν η συνθήκη είναι False εκτελούνται οι εντολές μέσα στο else και έπειτα οι (επόμενες εντολές)

Προσοχή:

- Μετα το else πρέπει να υπάρχει η άνω κάτω τελεία
- Με τους ίδιους κανόνες στοίχισης όπως στο if.

Παράδειγμα 4: comparison.py

```
age = 17

if age < 18:
    print("underage")
else:
    print("adult")
```

Παράδειγμα 5: odd.even.py

```
x = 3

if x % 2 == 0:
    print("even")
else:
    print("odd")
```

Άσκηση 4: Μέγιστος από 2

- Γράψτε ένα πρόγραμμα το οποίο θα δέχεται από την είσοδο δύο ακεραίους και έπειτα θα τυπώνει τον μέγιστο από αυτούς

Άσκηση 5: Μέγιστος από 3

- Γράψτε ένα πρόγραμμα το οποίο θα δέχεται από την είσοδο τρεις ακεραίους και έπειτα θα τυπώνει τον μέγιστο από αυτούς

ΣΥΜΒΟΥΛΕΥΘΕΙΤΕ απ' ευθείας το βίντεο σε αυτήν την άσκηση

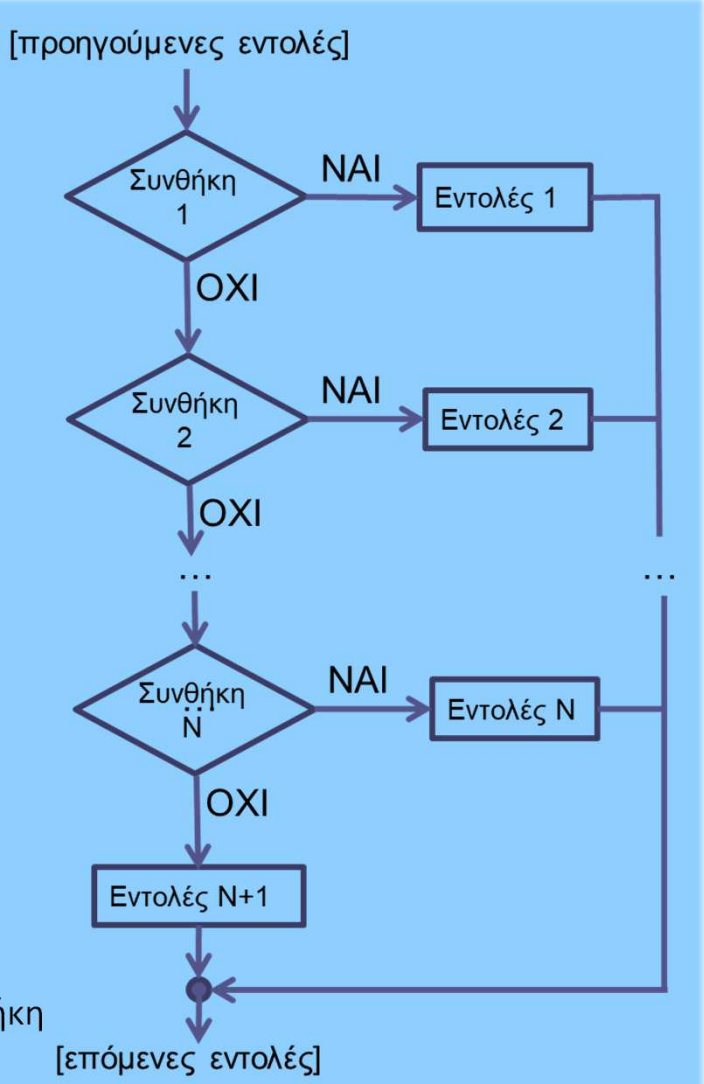
if…elif…else

- Συντακτικό:

```
[προηγούμενες εντολές]

if συνθήκη1:
   εντολές-1
elif συνθήκη2:
   εντολές-2
…
elif συνθήκηN:
   εντολές-N
else:
   εντολές-else

[επόμενες εντολές]
```



- Εξήγηση:
  - Αφού εκτελεστούν οι προηγούμενες εντολές
  - Αν επιτύχει η 1η συνθήκη τότε τρέχουν οι εντολές-1
  - Αλλιώς Αν επιτύχει η 2η συνθήκη τότε τρέχουν οι εντολές-N
  - …
  - Αλλιώς (έχουν αποτύχει όλες οι συνθήκες. Τρέχουν οι εντολές-else

Προσοχή:

- Μετα τη συνθήκη του elif (else if = αλλιώς αν) πρέπει να υπάρχει η άνω κάτω τελεία
- Με τους ίδιους κανόνες στοίχισης όπως στο if.

Παρατήρηση:

- Ενώ η if…else είναι για δύο περιπτώσεις
- Η if…elif…else γενικεύει για οσεσδήποτε περιπτώσεις.

Παράδειγμα 6: comparison2.py

```
a = 5
b = 3

if a > b:
   print("a>b")
elif a == b:
   print("a==b")
else:
   print("a<b")
```

Άσκηση 6: Διόρθωση στο πρόγραμμα του μέγιστου από 3

- Διορθώστε το πρόγραμμα της άσκησης 5, ώστε να δουλεύει σωστά και στην περίπτωση που ο μέγιστος είναι ίσος με κάποιον άλλο αριθμό.

Άσκηση 7: Τα στάδια της ζωής

- Γράψτε ένα πρόγραμμα το οποίο θα δέχεται από την είσοδο την ηλικία του χρήστη και θα τυπώνει μήνυμα για το αν είναι ανήλικός (<18), ενήλικός (18..65) ή συνταξιούχος (>65)

Άσκηση 8: Τάβλι

- Δύο έμπειροι παίκτες στο τάβλι, αποφασίζουν ποιος θα παίξει πρώτος, ρίχνοντας από δύο ζάρια ο καθένας. Νικάει αυτός που έχει το μεγαλύτερο άθροισμα στα δύο ζάρια.
- Κατασκευάστε ένα πρόγραμμα το οποίο θα αποφασίζει ποιος παίκτης νίκησε και θα παίξει πρώτος.

Παρατήρηση 1: Προαιρετικά κομμάτια στην if

- Προσοχή ότι σε μία εντολή if…elif… else τα κομμάτια elif και else είναι προαιρετικά (τα χρησιμοποιούμε αν τα χρειαζόμαστε).

```
if συνθήκη-if:
   εντολές-if
[elif συνθήκη-elif:
   εντολές-elif ]
[else:
   εντολές-else]
```

- Οι αγκύλες υποδεικνύουν ότι το κομμάτι που περικλείεται σε αυτές είναι προαιρετικό.

Παρατήρηση 2: Περισσότερα για τις συνθήκες

- Μία συνθήκη ερμηνεύεται ως True/False
- Ειδικά για τις λογικές μεταβλητές είναι σύνηθες να χρησιμοποιούνται χωρίς έλεγχο τιμής αλλά απ'ευθείας.
- Π.χ. αντί να γράφουμε για τη λογική μεταβλητή tired:

```
if tired == True:
```

προτιμάμε να γράφουμε:

```
if tired:
```

Παρατήρηση 3: Nested if

- Μία δομή ελέγχου που έχουμε συντάξει αποτελεί απλά μια ακόμη εντολή (συχνά την αναφέρουμε και σαν εντολή απόφασης)
- Έτσι, μπορεί να ενσωματωθεί σε οποιοδήποτε σημείο του προγράμματος γράφουμε εντολές.
- Και μπορούμε να έχουμε στις εντολές μιας δομής ελέγχου μια εντολή απόφασης και να έχουμε εμφωλευμένες if (nested if)

Παράδειγμα 7: nested.if.py

```
day = "Sunday"
tired = True

if day == "Saturday":
   print("I read a bit")
elif day == "Sunday":
   if tired:
      print("I won't study at all")
   else:
      print("I will study a lot")
else:
   print("I will study")
```

Παρατήρηση 4: Συμπτήξεις των εντολών απόφασης

- Μία απλή if που έχει μόνο μία εντολή:

```
if συνθήκη-if:
   εντολή-if
```

μπορεί να γραφεί και συντομογραφικά σε μία γραμμή:

```
if συνθήκη-if: εντολή-if
```

- Ενώ μία if-else που και τα δύο μέρη έχουν μόνο μία εντολή:

```
if συνθήκη:
   εντολή-if
else:
   εντολή-else
```

μπορεί να γραφεί συντομογραφικά σε μία γραμμή:

```
εντολή-if if συνθήκη-if  else: εντολή-else
```

- Αν και δεν θα χρησιμοποιήσουμε σε αυτά τα μαθήματα τις συμπτήξεις, καλό θα είναι να τις γνωρίζουμε, σε περίπτωση που τις συναντήσουμε σε κώδικες που θα αντιμετωπίσουμε.

Παρατήρηση 5: Η εντολή pass

- Η εντολή pass δεν κάνει τίποτα απολύτως (!)
- Είναι χρήσιμη π.χ. όταν γράφουμε μια δομή ελέγχου και θέλουμε να ελέγξουμε ένα μέρος της, χωρίς να έχουμε συμπληρώσει ακόμη όλη την εντολή.

```
if condition:
   pass
else:
   some working code..
```

Παράδειγμα 8: short.hand.if.py

```
number = 5

result = "odd" if number % 2 == 1 else "even"

print(result)
```

Σημείωση: Το αποτέλεσμα της συντομογραφίας της if-else “επιστρέφεται” και μπορούμε να το αποθηκεύσουμε σε μεταβλητή.

Άσκηση 9: Μια διόρθωση στην απεικόνιση της ώρας

Συνεχίζοντας την άσκηση 10 του προηγούμενου μαθήματος, διορθώστε το πρόγραμμα ώστε να απεικονίζει κομψα την ακριβή ώρα (π.χ. 01:14:08 αντί για 1:14:8)

Άσκηση 10: Το αγαπημένο φρούτο

Κατασκευάστε πρόγραμμα το οποίο θα προτρέπει το χρήστη να εισάγει το αγαπημένο του φρούτο.

- Αν ο χρήστης εισάγει το πορτοκάλι τότε το πρόγραμμα να απαντάει ότι όντως είναι υπέροχο
- Αν ο χρήστης εισάγει το μήλο το πρόγραμμα να απανταεί ότι είναι απαίσιο
- Αν ο χρήστης εισάγει οτιδήποτε άλλο, το πρόγραμμα να απαντάει ότι είναι απλά ικανοποιητικό.

Άσκηση 11: Πρωτοβάθμια Εξίσωση

- Γράψτε ένα πρόγραμμα που ζητάει από τον χρήστη να εισάγει 2 αριθμούς a,b και υπολογίζει την λύση της εξίσωσης ax+b=0 (Τα α και β είναι πραγματικοί αριθμοί)
- Υπενθύμιση: Αν το α δεν είναι 0 η λύση της εξίσωσης είναι x=-b/a. Αν το α είναι 0 να εκτυπώνεται το μήνυμα «Η εξίσωση δεν έχει λύση»

"Simple is better than complex."

Zen of Python #3