Πρόγραμμα Μεταπτυχιακών Σπουδών
Πληροφορική και Υπολογιστική Βιοϊατρική

Θέματα Προγραμματισμού Η/Υ

Ενότητα 6:

Θεματική Ενότητα: Λογικοί Τελεστές – Έλεγχος ροής

ΘΕΜΑΤΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Η/Υ

Θεματική Ενότητα 6

Λογικοί Τελεστές - Έλεγχος ροής

Πληροφορική και Υπολογιστική Βιοϊατρική
Α. Κακαρούντας, Γ. Σπαθούλας, Π. Κοντού

ΕΛΕΓΧΟΣ ΡΟΗΣ ΠΡΟΓΡΑΜΜΑΤΟΣ

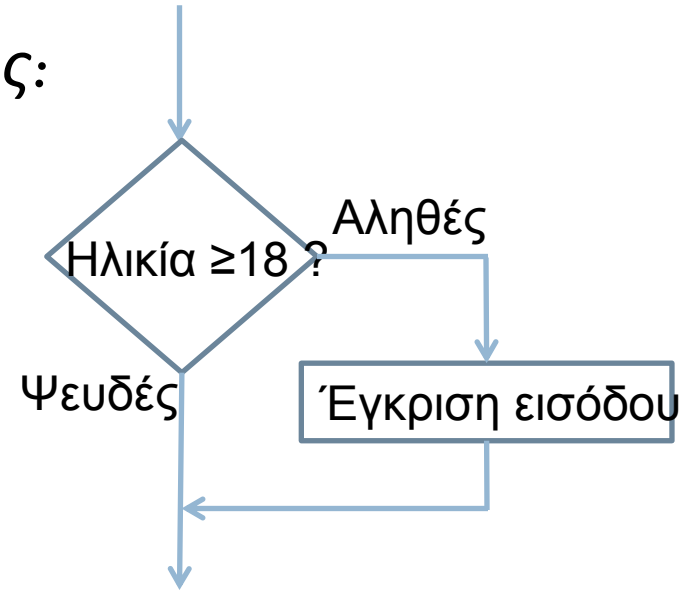
- Ελεγκτές συνθηκών ή περιπτώσεων (δομές ελέγχου):
  - if / if-else / if-elif-else
- Επαναληπτικές διαδικασίες (δομές επανάληψης):
  - for
  - while (με έλεγχο συνθήκης)

if

- Η εντολή if χρησιμοποιείται για τη δημιουργία δομών απόφασης, οι οποίες επιτρέπουν σε ένα πρόγραμμα να έχει πολλαπλές διαδρομές εκτέλεσης.
- Η εντολή if επιτρέπει σε μία ή περισσότερες εντολές να εκτελεστούν μόνο εάν κάποια λογική (Boolean) έκφραση είναι αληθής.
- *Δομή απόφασης μίας εναλλακτικής:*

```
if συνθήκη:
    εντολή 1
    εντολή 2
    ...
...
```

Μπλοκ εντολών



Λογικές Εκφράσεις και Σχεσιακοί Τελεστές

- Οι εκφράσεις που ελέγχονται από εντολές if ονομάζονται *λογικές εκφράσεις ή Boolean εκφράσεις, με αποτέλεσμα αληθές ή ψευδές.*
- Συνήθως, η λογική έκφραση σχηματίζεται με ένα *σχεσιακό τελεστή* ο οποίος *προσδιορίζει την ύπαρξη ή μη κάποιας συγκεκριμένης σχέσης* μεταξύ δύο τιμών.

Τελεστής	Έννοια
>	Μεγαλύτερο
<	Μικρότερο
> =	Μεγαλύτερο ή ίσο
< =	Μικρότερο ή ίσο
= =	Ίσο
! =	Διάφορο

π.χ.

```
>>> x = 1
>>> y = 0
>>> x > y
True
>>> x <= y
 False
>>> x == y
 False
```

```
>>> if x>=y:
        y=5
>>> y
5
>>> if x>=y:
        y=9
???
```

Λογικοί Τελεστές

- Συνδέουν δύο λογικές εκφράσεις σε μία σύνθετη έκφραση:
- έκφραση **and** έκφραση
  - Για να είναι αληθής η σύνθετη έκφραση, θα πρέπει και οι δύο υπο-εκφράσεις να είναι αληθείς.
- έκφραση **or** έκφραση
  - Για να είναι αληθής η σύνθετη έκφραση, θα πρέπει η μία ή και οι δύο υπο-εκφράσεις να είναι αληθείς. Αρκεί δηλαδή η μία (οποιαδήποτε) από τις δύο υπο-εκφράσεις να είναι αληθής.
- Μοναδιαίος τελεστής, εφαρμόζεται σε έναν μόνο τελεστέο:
- **not** (τελεστέος)
  - Ο τελεστέος θα πρέπει να είναι μία λογική έκφραση.
  - Ο τελεστής not αντιστρέφει την τιμή αληθείας του τελεστέου του.

Παραδείγματα Λογικών Τελεστών

- Αρχικές τιμές: a = 2, b = 4, c = 6.
- 6 <= c and a > 3 Ψ
- a >= -1 or a <= b Α
- not (a > 2) Α

- Έλεγχος Αριθμητικών Διαστημάτων Τιμών:
  - if x >= 20 and x <= 40 :

    print('Η τιμή είναι ***εντός*** του διαστήματος [20,40].')
  - if x <= 20 or x >= 40 :

    print('Η τιμή είναι ***εκτός*** του διαστήματος [20,40].')

Λογικές (Boolean) Μεταβλητές

```
>>> a=True
>>> type(a)
<class 'bool'>
>>> b=(2==3)
>>> b
False
>>>
```

- Εκτός από απευθείας αρχικοποίηση σε True ή False, παίρνουν συνήθως τιμή από αποτέλεσμα λογικών εκφράσεων*.
- Χρησιμοποιούνται συνήθως ως "σηματοδότες" ή "σημαίες" (flags).
  - Μία *flag* μεταβλητή σηματοδοτεί την ύπαρξη κάποιας συνθήκης στον κώδικα.
    - Όταν το flag είναι False, σημαίνει ότι η συνθήκη δεν ισχύει
    - Όταν το flag είναι True, σημαίνει ότι η συνθήκη ισχύει.

* Τιμές ≠0 αντιστοιχούν σε boolean True, 0 αντιστοιχεί σε boolean False

if-else

- Μία εντολή if-else θα εκτελέσει ένα μπλοκ εντολών εάν η συνθήκη της είναι αληθής ή ένα άλλο μπλοκ εάν η συνθήκη της είναι ψευδής.
- *Δομή απόφασης διπλής εναλλακτικής:*

if *συνθήκη:*

εντολή 1
εντολή 2
…

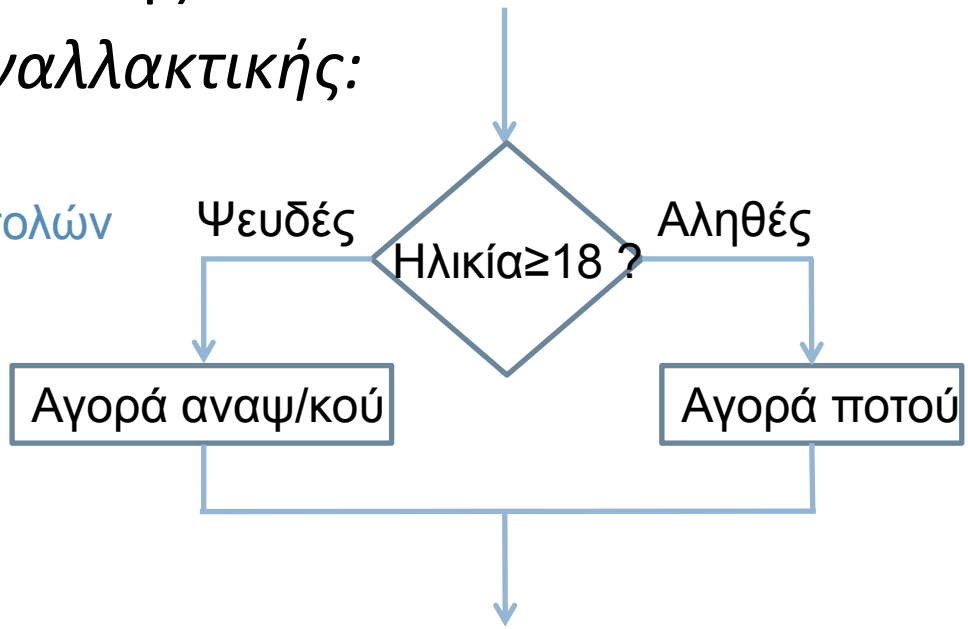
→ 1ο μπλοκ εντολών

else:

εντολή 1
εντολή 2
…

→ 2ο μπλοκ εντολών

…




```
if age >= 18:
        print("Επιτρέπεται το αλκοόλ")
else:
        print("Απαγορεύεται το
```

if-else παράδειγμα (1)

- Υπολογισμός μισθού προσαυξημένο με τις υπερωρίες
- Αλγόριθμος:
  - *Διάβασε τον αριθμό ωρών εργασίας*
  - *Διάβασε το ωρομίσθιο*
  - *Αν ο υπάλληλος δούλεψε περισσότερες από 40 ώρες:*
    - *Υπολόγισε και εμφάνισε τον μισθό με τις υπερωρίες*
  - *Διαφορετικά:*
    - *Υπολόγισε και εμφάνισε τον συνηθισμένο μισθό.*

if-else παράδειγμα (2)

```
# Μεταβλητές για τις βασικές ώρες εργασίας και τον πολλαπλασιαστή υπερωριών.
base_hours = 40                 # Βασικές ώρες την εβδομάδα
ot_multiplier = 1.5             # Πολλαπλασιαστής ωρομισθίου υπερωριών (+50%)

# Εισαγωγή των ωρών εργασίας και του ωρομισθίου.
hours = float( input('Δώσε το αριθμό των ωρών εργασίας: ') )
pay_rate = float( input('Δώσε το ωρομίσθιο: ') )

# Υπολογισμός και εμφάνιση του μισθού.
if hours > base_hours:
    # Υπολογισμός του μισθού με τις υπερωρίες.
    overtime_hours = hours - base_hours          # Υπολογισμός αριθμού υπερωριών.

    overtime_pay = overtime_hours * pay_rate * ot_multiplier    # Ποσό πληρωμής υπερωριών.

    gross_pay = base_hours * pay_rate + overtime_pay          # Συνολικός μισθός.
else
    # Υπολογισμός του μισθού χωρίς υπερωρίες.
    gross_pay = hours * pay_rate

print('Ο μισθός σας είναι  €', format(gross_pay, '.2f'), sep='' )    # Εμφάνιση συνολικού μισθού.
```

```
Δώσε το αριθμό των ωρών εργασίας: 40
Δώσε το ωρομίσθιο: 20
Ο μισθός σας είναι €800.00
```

```
Δώσε το αριθμό των ωρών εργασίας: 50
Δώσε το ωρομίσθιο: 20
Ο μισθός σας είναι €1,100.00
```

Σύγκριση συμβολοσειρών (1)

- = =, !=

```
your_name = input('Δώσε το όνομά σου: ')
user_name = 'Kostas'
if your_name == user_name :  #Case sensitive σύγκριση (μπορεί να αλλάξει)
    print('Απόκτηση πρόσβασης')
else :
    print('Απόρριψη πρόσβασης')
```

- >, <, >=, <=

```
if 'a' < 'b' :        # σύγκριση του ascii κωδικού τους
    print('a μικρότερο του b')
else :
    print('a μεγαλύτερο του b')
```

Σύγκριση συμβολοσειρών (2)

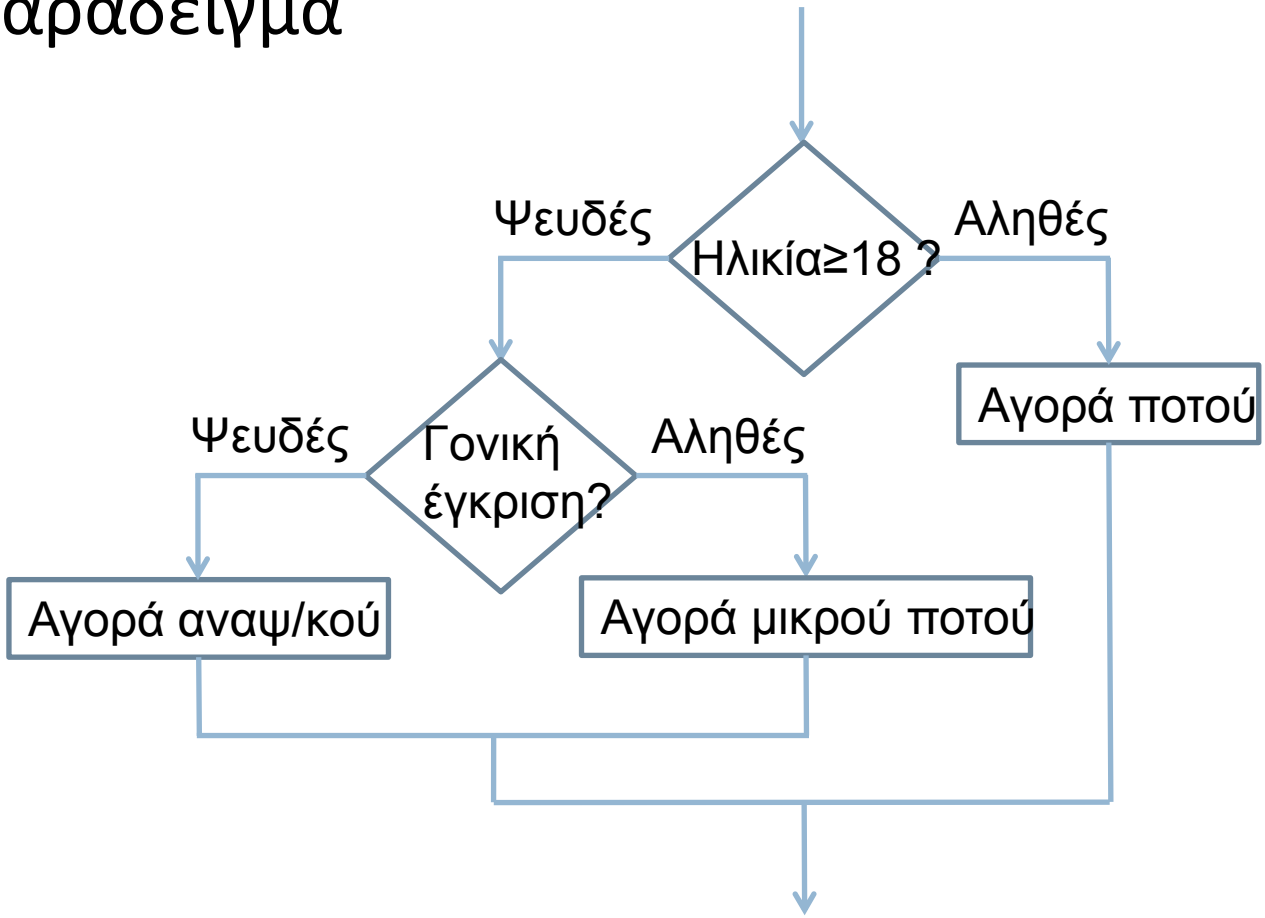
- Σύγκριση συμβολοσειρών πολλών χαρακτήρων
  - Σύγκριση ένα προς ένα των χαρακτήρων
  - Στην πρώτη διαφοροποίηση χαρακτήρα μικρότερος είναι αυτός με τον μικρότερο ascii κωδικό (αλφαβητικά – κενό, αριθμοί, κεφαλαία, μικρά)
  - Για διαφορετικού μήκους συμβολοσειρές που μέχρι το τέλος της μικρότερης συμπίπτουν (π.χ. "Hello" και "Hell") μικρότερη είναι αυτή με το μικρότερο μήκος.

```
name1 = 'Marios'
name2 = 'Markos'
if name1 > name2:
    print('Το', name1, 'είναι μεγαλύτερο του', name2)
else:
    print('Το', name2, 'είναι μεγαλύτερο του', name1)
```

Χρήση για αλφαβητική ταξινόμηση ονομάτων

Ένθετες Δομές Απόφασης

- Παράδειγμα



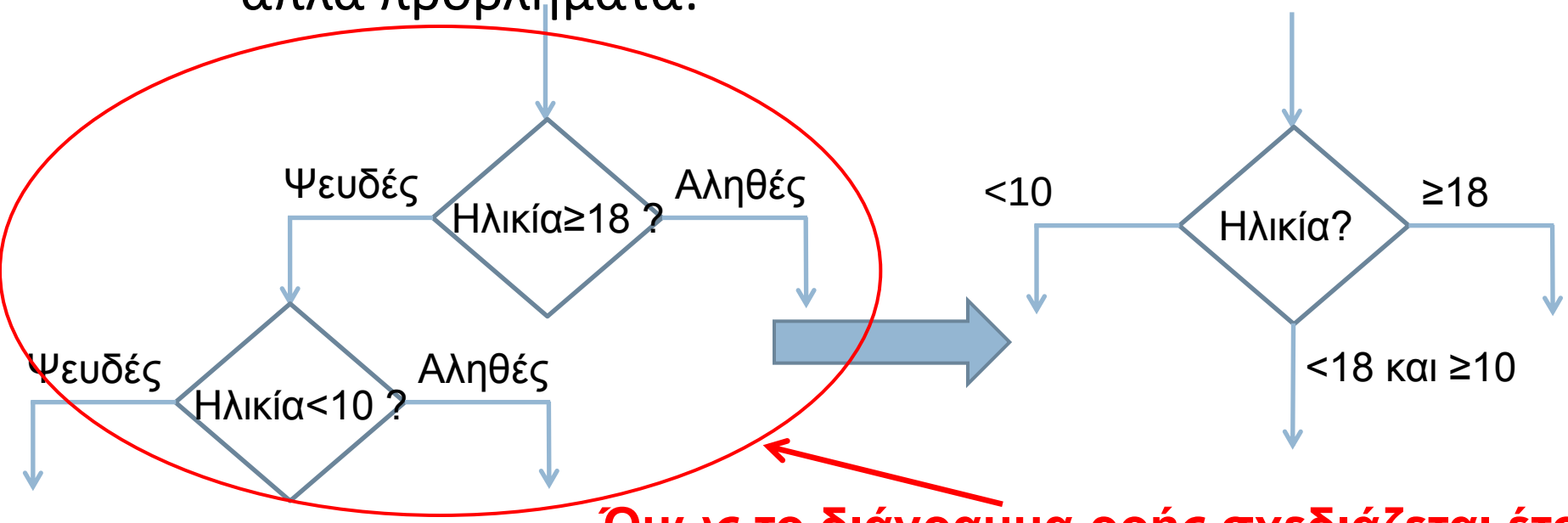
Ένθετες Δομές Απόφασης - κώδικας

```
age = int( input('Δώσε την ηλικία σου: ')
goniki_egrisi = input('Έχεις γονική έγκριση? Δώσε Ν για ναι ή Ο για όχι: ')
if goniki_egrisi == 'N' :
    goniki_egrisi = True                    # Boolean μεταβλητή
else :
     goniki_egrisi = False

if age >=18 :
    print('Μπορείς να αγοράσεις ποτό')
else :
    if goniki_egrisi == True :
            print('Μπορείς να αγοράσεις μικρό ποτό')
    else :
            print('Μπορείς να αγοράσεις αναψυκτικό')
```

if-elif-else

- Επέκταση της if-else λογικής για παραπάνω των 2 εναλλακτικών επιλογών (true/false τις ίδιας λογικής έκφρασης)
  - Η χρήση ένθετων δομών δίνει πολύπλοκες λύσεις σε απλά προβλήματα:



Όμως το διάγραμμα ροής σχεδιάζεται έτσι!

if-elif-else Παράδειγμα

- Γράψτε ένα πρόγραμμα που να ζητάει από το χρήστη την ηλικία του και θα εμφανίζει ένα μήνυμα που να αναφέρει ένα το άτομο είναι βρέφος, παιδί, έφηβος ή ενήλικας, σύμφωνα με τα διαστήματα:
  - Αν είναι 1 χρόνου ή μικρότερο, είναι βρέφος.
  - Αν είναι μεγαλύτερο από 1 αλλά μικρότερο από 13, είναι παιδί.
  - Αν είναι τουλάχιστον 13 αλλά μικρότερο από 20, είναι έφηβος.
  - Αν είναι τουλάχιστον 20 χρονών, είναι ενήλικας.

```
age = float(input("Δώσε την ηλικία σου σε πραγματικό αριθμό (θετικό): ")
if age <= 1 :
    print("Είσαι βρέφος!")
elif age >1 and age<13 :
     print("Είσαι παιδί!")
elif age >=13 and age<20 :
     print("Είσαι έφηβος!")
else :
     print("Είσαι ενήλικας!")
```

ΘΕΜΑΤΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Η/Υ

Θεματική Ενότητα 6

Λογικοί Τελεστές - Έλεγχος ροής

Πληροφορική και Υπολογιστική Βιοϊατρική
Α. Κακαρούντας, Γ. Σπαθούλας, Π. Κοντού