ΕΝΟΤΗΤΑ 2

ΟΡΙΣΜΟΣ ΠΡΟΒΛΗΜΑΤΟΣ

Πρόβλημα είναι μια κατάσταση η οποία πρέπει να αντιμετωπιστεί, απαιτεί λύση, η δε λύση της δεν είναι γνωστή, ούτε προφανής.

ΚΑΤΗΓΟΡΙΕΣ ΠΡΟΒΛΗΜΑΤΩΝ

- Οικονομικά
- Υγείας
- Περιβαλλοντικά
- Αριθμητικά
- Κοινωνικά
- Πολιτικά

ΚΑΤΗΓΟΡΙΕΣ ΠΡΟΒΛΗΜΑΤΩΝ ΩΣ ΠΡΟΣ ΤΗ ΔΥΝΑΤΟΤΗΤΑ ΕΠΙΛΥΣΗΣ ΤΟΥΣ

1. Επιλύσιμα, δηλαδή αυτά για τα οποία η λύση τους έχει βρεθεί και διατυπωθεί, π.χ. η εξίσωση μιας δευτεροβάθμιας εξίσωσης.
2. Μη επιλύσιμα, δηλαδή αυτά για τα οποία έχει αποδειχτεί ότι δεν μπορούν να λυθούν π.χ. ο τετραγωνισμός του κύκλου.
3. Ανοιχτά, δηλαδή αυτά για τα οποία η λύση τους δεν έχει βρεθεί ακόμα ενώ ταυτόχρονα δεν έχει αποδειχθεί ότι δεν μπορούν να επιλυθούν π.χ. η ενοποίηση των τεσσάρων πεδίων δυνάμεων.

ΥΠΟΛΟΓΙΣΤΙΚΟ ΠΡΟΒΛΗΜΑ

Οποιοδήποτε πρόβλημα μπορεί να επιλυθεί και με τη χρήση ηλεκτρονικού υπολογιστή λέγεται υπολογιστικό πρόβλημα.

ΔΕΔΟΜΕΝΟ, ΠΛΗΡΟΦΟΡΙΑ, ΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ

Με τον όρο **δεδομένο** δηλώνεται οποιοδήποτε στοιχείο μπορεί να γίνει αντιληπτό από έναν τουλάχιστον παρατηρητή με μία από τις πέντε αισθήσεις του.
Με τον όρο **πληροφορία** αναφέρεται οποιοδήποτε γνωσιακό στοιχείο προέρχεται από επεξεργασία δεδομένων.
Ο όρος **επεξεργασία δεδομένων** δηλώνει εκείνη τη διαδικασία κατά την οποία ένας "μηχανισμός" δέχεται δεδομένα, τα επεξεργάζεται σύμφωνα με έναν προκαθορισμένο τρόπο και αποδίδει πληροφορίες.

Επί χιλιετίες ο "μηχανισμός" επεξεργασίας των δεδομένων ήταν και εξακολουθεί να είναι ο ανθρώπινος εγκέφαλος. Στις μέρες μας, ένας άλλος "μηχανισμός" επεξεργασίας δεδομένων είναι ο υπολογιστής.

ΣΤΑΔΙΑ ΑΝΤΙΜΕΤΩΠΙΣΗΣ ΠΡΟΒΛΗΜΑΤΟΣ

Τα στάδια αντιμετώπισης ενός προβλήματος είναι τρία:

- κατανόηση, όπου απαιτείται η σωστή και πλήρης αποσαφήνιση των δεδομένων και των ζητούμενων του προβλήματος
- ανάλυση, όπου το αρχικό πρόβλημα διασπάται σε άλλα επιμέρους απλούστερα προβλήματα
- επίλυση, όπου υλοποιείται η λύση του προβλήματος, μέσω της λύσης των επιμέρους προβλημάτων.

ΟΡΙΣΜΟΣ ΑΛΓΟΡΙΘΜΟΥ

Είναι μια πεπερασμένη σειρά ενεργειών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος.

ΚΡΙΤΗΡΙΑ ΠΟΥ ΠΡΕΠΕΙ ΝΑ ΙΚΑΝΟΠΟΙΕΙ ΚΑΘΕ ΑΛΓΟΡΙΘΜΟΣ

- ✓ **ΕΙΣΟΔΟΣ** (καμία, μία ή περισσότερες τιμές)
- ✓ **ΕΞΟΔΟΣ** (τουλάχιστον μια τιμή)
- ✓ **ΚΑΘΟΡΙΣΤΙΚΟΤΗΤΑ** (κάθε εντολή πρέπει να καθορίζεται χωρίς καμιά αμφιβολία για τον τρόπο εκτέλεσής της)
- ✓ **ΠΕΡΑΤΟΤΗΤΑ** (πρέπει να τελειώνει μετά από πεπερασμένα βήματα εκτέλεσης των εντολών του)
- ✓ **ΑΠΟΤΕΛΕΣΜΑΤΙΚΟΤΗΤΑ** (κάθε μια εντολή πρέπει να είναι απλή και εκτελέσιμη)

ΑΝΑΠΑΡΑΣΤΑΣΗ ΑΛΓΟΡΙΘΜΩΝ

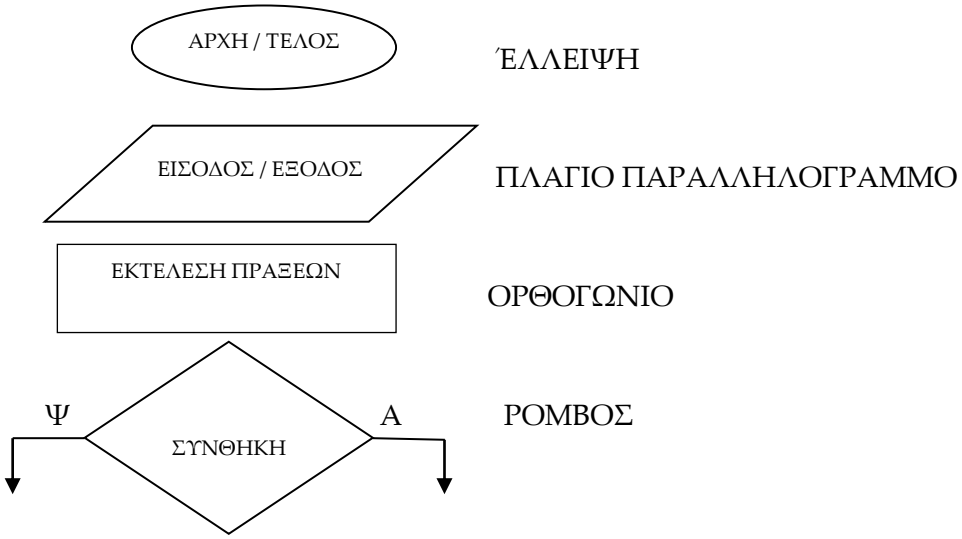
Γίνεται με τους παρακάτω τρόπους:

- ✓ **ΕΛΕΥΘΕΡΟ ΚΕΙΜΕΝΟ**
- ✓ **ΔΙΑΓΡΑΜΜΑΤΙΚΕΣ ΤΕΧΝΙΚΕΣ (διάγραμμα ροής)**
- ✓ **ΦΥΣΙΚΗ ΓΛΩΣΣΑ ΚΑΤΑ ΒΗΜΑΤΑ**
- ✓ **ΚΩΔΙΚΟΠΟΙΗΣΗ (ψευδογλώσσα)**

Εμείς θα σχεδιάζουμε αλγορίθμους με διαγραμματικές τεχνικές και κωδικοποίηση.

ΔΙΑΓΡΑΜΜΑΤΑ ΡΟΗΣ

Χρησιμοποιούν τα παρακάτω σχήματα:



ΈΛΛΕΙΨΗ

ΠΛΑΓΙΟ ΠΑΡΑΛΛΗΛΟΓΡΑΜΜΟ

ΟΡΘΟΓΩΝΙΟ

ΡΟΜΒΟΣ

ΠΩΣ ΔΗΜΙΟΥΡΓΟΥΜΕ ΑΛΓΟΡΙΘΜΟ ΜΕ ΚΩΔΙΚΟΠΟΙΗΣΗ (ΨΕΥΔΟΓΛΩΣΣΑ)

1.Κάθε αλγόριθμος ξεκινάει με την λέξη **ΑΛΓΟΡΙΘΜΟΣ** ακολουθούμενη από ένα όνομα που του δίνουμε.

- Το όνομα που δίνουμε μπορεί να αποτελείται από γράμματα της ελληνικής ή λατινικής αλφαβήτου κεφαλαία ή πεζά και αριθμούς.
- Το όνομα **ΔΕΝ** πρέπει να ξεκινάει με αριθμό , π.χ. δεν είναι σωστό να γράψουμε ΑΛΓΟΡΙΘΜΟΣ 1ΑΣΚΗΣΗ. Το σωστό θα ήταν να γράψουμε ΑΛΓΟΡΙΘΜΟΣ ΑΣΚΗΣΗ1
- Αν το όνομα του αλγορίθμου αποτελείται από περισσότερες από μία λέξη **ΔΕΝ** επιτρέπεται να αφήνουμε κενό ανάμεσα σε αυτές αλλά χρησιμοποιούμε σαν διαχωριστικό την κάτω παύλα (_) π.χ. δεν είναι σωστό να γράψουμε ΠΡΟΓΡΑΜΜΑ Γ ΤΑΞΗ ΛΥΚΕΙΟΥ. Το σωστό θα ήταν να γράψουμε ΠΡΟΓΡΑΜΜΑ Γ_ΤΑΞΗ_ΛΥΚΕΙΟΥ. Το όνομα δεν πρέπει να περιέχει άλλα σύμβολα, το μόνο επιτρεπτό σύμβολο είναι η κάτω παύλα.
- Δεν επιτρέπεται να χρησιμοποιούμε σαν ονόματα μεταβλητών κάποιες «δεσμευμένες» λέξεις όπως είναι τα ονόματα εντολών π.χ. ΑΛΓΟΡΙΘΜΟΣ, ΔΙΑΒΑΣΕ, ΕΚΤΥΠΩΣΕ

2.Κάθε αλγόριθμος χρησιμοποιεί τις **ΜΕΤΑΒΛΗΤΕΣ** δηλ. μεγέθη η τιμή των οποίων μπορεί να αλλάξει κατά τη διάρκεια εκτέλεσης του αλγορίθμου και τις **ΣΤΑΘΕΡΕΣ** δηλ. μεγέθη των οποίων η τιμή παραμένει αμετάβλητη κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Οι μεταβλητές και οι σταθερές έχουν κι αυτές ονόματα που τις δίνουμε εμείς και για τα ονόματά τους ισχύουν οι περιορισμοί που προαναφέραμε παραπάνω για τα ονόματα των αλγορίθμων.

3. Οι **ΣΤΑΘΕΡΕΣ** και οι **ΜΕΤΑΒΛΗΤΕΣ** διακρίνονται σε 4 κατηγορίες ανάλογα με το είδος των τιμών που μπορούν να πάρουν:

- **ΑΚΕΡΑΙΕΣ** π.χ 0, 15, -25
- **ΠΡΑΓΜΑΤΙΚΕΣ** π.χ 0.2, 12.12, -3.966
- **ΧΑΡΑΚΤΗΡΕΣ** π.χ. ‘μ’, ‘μπαταρία’, ‘15’. Οι μεταβλητές τύπου χαρακτήρα μπαίνουν πάντοτε μέσα σε μονά εισαγωγικά ‘…’
- **ΛΟΓΙΚΕΣ** όταν παίρνουν δυο τιμές ΑΛΗΘΗΣ (TRUE) ή ΨΕΥΔΗΣ (FALSE)

4.Στις επόμενες σειρές γράφουμε τις **ΕΝΤΟΛΕΣ.** Οι ΕΝΤΟΛΕΣ χωρίζονται σε τρείς κατηγορίες:

- **ΕΝΤΟΛΗ ΕΙΣΟΔΟΥ** που είναι η **ΔΙΑΒΑΣΕ**
- **ΕΝΤΟΛΗ ΕΞΟΔΟΥ** που είναι η **ΓΡΑΨΕ**, η **ΕΚΤΥΠΩΣΕ** και η **ΕΜΦΑΝΙΣΕ**
- **ΕΝΤΟΛΗ ΕΚΧΩΡΗΣΗΣ** όπου εκχωρούμε σε μια μεταβλητή μια τιμή ή το αποτέλεσμα μιας πράξης. Κατά την εκχώρηση η μεταβλητή χάνει το περιεχόμενο που πιθανώς είχε πριν. Η εντολή εκχώρησης συντάσσετε με το σύμβολο ← .Αριστερά του συμβόλου υπάρχει **μόνο μια μεταβλητή** ενώ δεξιά μπορεί να υπάρχει αριθμός, μεταβλητή, μαθηματική παράσταση.

5.Τέλος, ο αλγόριθμος κλείνει **ΠΑΝΤΑ** με την εντολή **ΤΕΛΟΣ** και δίπλα το όνομά του.

Μια γενική λοιπόν δομή ενός αλγορίθμου είναι η ακόλουθη:

```
ΑΛΓΟΡΙΘΜΟΣ ΑΣΚΗΣΗ_1
        ΔΙΑΒΑΣΕ ΜΕΤΑΒΛΗΤΗ_1, ΜΕΤΑΒΛΗΤΗ_2
        ΜΕΤΑΒΛΗΤΗ_3← ΜΕΤΑΒΛΗΤΗ_1 + ΜΕΤΑΒΛΗΤΗ_2
        ΓΡΑΨΕ ΜΕΤΑΒΛΗΤΗ_3
ΤΕΛΟΣ ΑΣΚΗΣΗ_1
```

6.Τα σύμβολα (τελεστές) των αριθμητικών πράξεων που εκτελούν οι αλγόριθμοι είναι τα παρακάτω:

ΠΡΟΣΘΕΣΗ	+
ΑΦΑΙΡΕΣΗ	-
ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ	*
ΔΙΑΙΡΕΣΗ	/
ΥΨΩΣΗ ΣΕ ΔΥΝΑΜΗ	^
ΠΗΛΙΚΟ ΑΚΕΡΑΙΗΣ ΔΙΑΙΡΕΣΗΣ	div
ΥΠΟΛΟΙΠΟ ΑΚΕΡΑΙΗΣ ΔΙΑΙΡΕΣΗΣ	mod

Όταν πρέπει να υπολογίσουμε σύνθετες μαθηματικές παραστάσεις κάνουμε σωστή χρήση παρενθέσεων με προτεραιότητα εκτέλεσης όπως γνωρίζουμε από τα Μαθηματικά.

ΔΟΜΕΣ ΑΛΓΟΡΙΘΜΩΝ

Υπάρχουν τρεις βασικές δομές που χρησιμοποιούμε στους αλγορίθμους και στα προγράμματα:

- ✓ ΔΟΜΗ ΑΚΟΛΟΥΘΙΑΣ
- ✓ ΔΟΜΗ ΕΠΙΛΟΓΗΣ
- ✓ ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ

ΔΟΜΗ ΑΚΟΛΟΥΘΙΑΣ

Είναι η πιο απλή δομή, κατά την οποία η σειρά εκτέλεσης ενός συνόλου ενεργειών είναι δεδομένη δηλαδή οι εντολές ακολουθούν η μια την άλλη.

ΔΟΜΗ ΕΠΙΛΟΓΗΣ

Η δομή της επιλογής περιλαμβάνει τον έλεγχο κάποιας **λογικής συνθήκης** που μπορεί να έχει δυο τιμές (ΑΛΗΘΗΣ ή ΨΕΥΔΗΣ) και ακολουθεί η απόφαση εκτέλεσης κάποιας ενέργειας με βάση την τιμή αυτής της λογικής συνθήκης π.χ. αν βρέχει θα πάρω μαζί μου ομπρέλα διαφορετικά δεν θα πάρω δηλ. αν η συνθήκη(βροχή) είναι ΑΛΗΘΗΣ θα πάρω μαζί μου ομπρέλα ενώ αν είναι ΨΕΥΔΗΣ δεν θα πάρω.

Οι **λογικές συνθήκες** χρησιμοποιούν συνήθως τους παρακάτω **τελεστές σύγκρισης**:

>	Μεγαλύτερο	Π.χ. Χ>10
<	Μικρότερο	Π.Χ Χ<Υ
>=	Μεγαλύτερο ή ίσο	Π.χ. Χ>=20
=<	Μικρότερο ή ίσο	Π.χ Χ=<0
=	Ίσο	Π.χ. Χ=Υ
<>	διάφορο	Π.χ Χ<>Υ

Όταν αριθμητικοί και συγκριτικοί τελεστές συνδυάζονται σε μια έκφραση, οι αριθμητικές πράξεις εκτελούνται πρώτες.

Υπάρχουν όμως και πιο σύνθετες λογικές συνθήκες, οι οποίες σχηματίζονται με την χρήση των παρακάτω **λογικών τελεστών:**

ΚΑΙ	ΣΥΖΕΥΞΗ	Π.χ Χ<10 **ΚΑΙ** Χ>20
Η	ΔΙΑΖΕΥΞΗ	Π.χ Χ<>0 **Η** Χ>3
ΟΧΙ	ΑΡΝΗΣΗ	Π.χ **ΟΧΙ** Χ>2

Έστω ότι έχουμε δυο συνθήκες, την ΣΥΝΘΗΚΗ Α και την ΣΥΝΘΗΚΗ Β. Ο παρακάτω πίνακας μας δείχνει τις τιμές των λογικών πράξεων για όλους τους συνδυασμούς τιμών:

ΣΥΝΘΗΚΗ Α	**ΣΥΝΘΗΚΗ Β**	Α **Η** Β	Α **ΚΑΙ** Β	**ΟΧΙ** Α
ΑΛΗΘΗΣ	ΑΛΗΘΗΣ	ΑΛΗΘΗΣ	ΑΛΗΘΗΣ	ΨΕΥΔΗΣ
ΑΛΗΘΗΣ	ΨΕΥΔΗΣ	ΑΛΗΘΗΣ	ΨΕΥΔΗΣ	ΨΕΥΔΗΣ
ΨΕΥΔΗΣ	ΑΛΗΘΗΣ	ΑΛΗΘΗΣ	ΨΕΥΔΗΣ	ΑΛΗΘΗΣ
ΨΕΥΔΗΣ	ΨΕΥΔΗΣ	ΨΕΥΔΗΣ	ΨΕΥΔΗΣ	ΑΛΗΘΗΣ

Δηλαδή η λογική πράξη **Η** είναι ΑΛΗΘΗΣ όταν οποιαδήποτε από τις δυο συνθήκες είναι ΑΛΗΘΗΣ.

Η λογική πράξη **ΚΑΙ** είναι ΑΛΗΘΗΣ μόνο όταν και οι δυο συνθήκες είναι ΑΛΗΘΕΙΣ.

Η λογική πράξη **ΟΧΙ** είναι ΑΛΗΘΗΣ όταν η πρόταση που την ακολουθεί είναι ΨΕΥΔΗΣ

Οι λογικοί τελεστές έχουν χαμηλότερη ιεραρχία από τους συγκριτικούς.

Η πιο απλή σύνταξη της δομής επιλογής είναι η ΑΠΛΗ ΕΠΙΛΟΓΗ:

ΑΝ <λογική συνθήκη> ΤΟΤΕ
ΕΝΤΟΛΗ/ΕΣ
ΤΕΛΟΣ_ΑΝ

ΠΑΡΑΔΕΙΓΜΑ:

```
ΑΝ Χ>0 ΤΟΤΕ
    ΓΡΑΨΕ 'ΘΕΤΙΚΟΣ'
ΤΕΛΟΣ_ΑΝ
```

Η γενική σύνταξη της σύνθετης δομή επιλογής είναι η ΣΥΝΘΕΤΗ ΕΠΙΛΟΓΗ:

ΑΝ <λογική συνθήκη> ΤΟΤΕ
ΟΜΑΔΑ ΕΝΤΟΛΩΝ1
ΑΛΛΙΩΣ
ΟΜΑΔΑ ΕΝΤΟΛΩΝ2
ΤΕΛΟΣ_ΑΝ

ΠΑΡΑΔΕΙΓΜΑ:

```
ΑΝ Χ>0 ΤΟΤΕ
        ΕΜΦΑΝΙΣΕ 'ΘΕΤΙΚΟΣ'
ΑΛΛΙΩΣ
        ΕΜΦΑΝΙΣΕ 'ΜΗ ΘΕΤΙΚΟΣ'
ΤΕΛΟΣ_ΑΝ
```

Υπάρχουν ωστόσο περιπτώσεις όπου μπορεί να ληφθούν παραπάνω από δυο διαφορετικές αποφάσεις ανάλογα με την τιμή που παίρνει μια έκφραση. Σε αυτές τις περιπτώσεις χρησιμοποιούμε την **ΠΟΛΛΑΠΛΗ ΕΠΙΛΟΓΗ.**

ΑΝ <λογική συνθήκη1> ΤΟΤΕ
ΟΜΑΔΑ ΕΝΤΟΛΩΝ1
ΑΛΛΙΩΣ_ΑΝ <λογική συνθήκη2> ΤΟΤΕ
ΟΜΑΔΑ ΕΝΤΟΛΩΝ2
ΑΛΛΙΩΣ_ΑΝ <λογική συνθήκη3> ΤΟΤΕ
ΟΜΑΔΑ ΕΝΤΟΛΩΝ3
.
.
ΑΛΛΙΩΣ
ΟΜΑΔΑ ΕΝΤΟΛΩΝν
ΤΕΛΟΣ_ΑΝ

ΠΑΡΑΔΕΙΓΜΑ

```
ΑΝ Χ>0 ΤΟΤΕ
        ΕΜΦΑΝΙΣΕ 'ΘΕΤΙΚΟΣ'
ΑΛΛΙΩΣ_ΑΝ Χ<0 ΤΟΤΕ
        ΕΜΦΑΝΙΣΕ 'ΑΡΝΗΤΙΚΟΣ'
ΑΛΛΙΩΣ
        ΕΜΦΑΝΙΣΕ 'ΜΗΔΕΝ'
ΤΕΛΟΣ_ΑΝ
```

Τέλος, μια άλλη μορφή που μπορεί να έχει η δομή επιλογής είναι αυτή της **εμφωλευμένης δομής,** όπου μια ή περισσότερες δομές επιλογής περιέχονται μέσα σε μια δομή επιλογής. Π.χ.

```
ΑΝ Χ>=10 ΤΟΤΕ
        ΑΝ Χ>18 ΤΟΤΕ
                ΓΡΑΨΕ 'ΠΑΙΡΝΕΙΣ ΑΡΙΣΤΕΙΟ'
        ΑΛΛΙΩΣ
                ΓΡΑΨΕ 'ΔΕΝ ΠΑΙΡΝΕΙΣ ΑΡΙΣΤΕΙΟ'
        ΤΕΛΟΣ_ΑΝ
ΑΛΛΙΩΣ
        ΓΡΑΨΕ 'ΕΠΑΝΑΛΑΜΒΑΝΕΙΣ ΤΗΝ ΤΑΞΗ'
ΤΕΛΟΣ_ΑΝ
```

ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ

Η δομή επανάληψης εφαρμόζεται όταν μια ομάδα εντολών πρέπει να εκτελεστεί πολλές φορές, ανάλογα με την τιμή μιας συνθήκης.
Υπάρχουν τρεις δομές επανάληψης:

1η ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ

ΟΣΟ <συνθήκη> ΕΠΑΝΑΛΑΒΕ
Εντολές
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

Οι εντολές εκτελούνται όσο η συνθήκη είναι αληθής. Η δομή επανάληψης τερματίζεται όταν η συνθήκη γίνει ψευδής(αυτό μπορεί να συμβεί από την αρχή κι η δομή επανάληψης να μην εκτελεστεί ούτε μια φορά)

2η ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ

ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ
Εντολές
ΜΕΧΡΙΣ_ΟΤΟΥ <συνθήκη>

Οι εντολές εκτελούνται μέχρι η συνθήκη να γίνει αληθής.

Οι διαφορές της δομής ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ …ΜΕΧΡΙΣ_ΟΤΟΥ με την δομή ΟΣΟ…ΕΠΑΝΑΛΑΒΕ είναι οι εξής:

- ✓ **Η δομή ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ …ΜΕΧΡΙΣ_ΟΤΟΥ εκτελείται τουλάχιστον μια φορά ενώ η δομή ΟΣΟ…ΕΠΑΝΑΛΑΒΕ υπάρχει περίπτωση να μην εκτελεστεί ούτε μια φορά**
- ✓ **Στη δομή ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ …ΜΕΧΡΙΣ_ΟΤΟΥ η συνθήκη βρίσκεται στο τέλος ενώ στη δομή ΟΣΟ…ΕΠΑΝΑΛΑΒΕ η συνθήκη βρίσκεται στην αρχή**
- ✓ **Η δομή ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ …ΜΕΧΡΙΣ_ΟΤΟΥ εκτελείται μέχρι η συνθήκη να γίνει αληθής ενώ η δομή ΟΣΟ…ΕΠΑΝΑΛΑΒΕ εκτελείται όσο η συνθήκη είναι αληθής.**

3η ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ

ΓΙΑ <μεταβλητή> ΑΠΟ <αρχική τιμή> ΜΕΧΡΙ <τελική τιμή> με_βήμα<μεταβολή τιμής>
Εντολές
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

Η δομή αυτή χρησιμοποιείται όταν γνωρίζουμε εκ των προτέρων τον αριθμό των επαναλήψεων.

Το βήμα μπορεί να είναι θετικός ή αρνητικός αριθμός και μας δείχνει κατά πόσο θα μεταβληθεί η τιμή της μεταβλητής. Το βήμα δεν μπορεί να είναι μηδέν γιατί ο αλγόριθμος δεν θα είχε νόημα εφόσον η επανάληψη δεν θα τερματιζόταν ποτέ.

ΠΙΝΑΚΕΣ

Ως πίνακα ορίζουμε μια δομή που περιέχει στοιχεία του ίδιου τύπου (ακέραιους, πραγματικούς αριθμούς, χαρακτήρες).
Ένας πίνακας μπορεί να είναι μονοδιάστατος, δισδιάστατος, τρισδιάστατος και γενικά ν-διάστατος.
Εμείς θα ασχοληθούμε με μονοδιάστατους και δισδιάστατους πίνακες.

ΜΟΝΟΔΙΑΣΤΑΤΟΣ ΠΙΝΑΚΑΣ

Παρακάτω φαίνεται ένας μονοδιάστατος πίνακας:

ι=1	ι=2	ι=3	ι=4	ι=5	ι=6	ι=7	ι=8	ι=9	ι=10

Παρατηρούμε ότι ο συγκεκριμένος πίνακας έχει 10 θέσεις.
Σε έναν αλγόριθμο ή σε ένα πρόγραμμα για να συμβολίσουμε έναν μονοδιάστατο πίνακα του δίνουμε ένα (επιτρεπτό) όνομα και μέσα σε αγκύλη τον αριθμό των θέσεών του. Έτσι, ο παραπάνω πίνακας θα μπορούσε να ονομαστεί table[10].
Για να διαβάσει τον παραπάνω πίνακα ένας αλγόριθμος ή ένα πρόγραμμα(δηλ. για να τον «γεμίσει» με δεδομένα) χρησιμοποιεί την επαναληπτική δομή ΓΙΑ…ΑΠΟ…ΜΕΧΡΙ. Έτσι έχω:

```
ΓΙΑ ι ΑΠΟ 1 ΜΕΧΡΙ 10
    ΔΙΑΒΑΣΕ table[ι]
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
```

όπου το ι είναι ο λεγόμενος «δείκτης», δηλ. μας δείχνει σε ποια θέση του πίνακα βρισκόμαστε.
Έστω ότι «γεμίζω» τον πίνακα με τους αριθμούς 10,20,30,40,50,60,70,80,90,100 οπότε γίνεται:

10	20	30	40	50	60	70	80	90	100
ι=1	ι=2	ι=3	ι=4	ι=5	ι=6	ι=7	ι=8	ι=9	ι=10

ΔΙΣΔΙΑΣΤΑΤΟΣ ΠΙΝΑΚΑΣ

Παρακάτω φαίνεται ένας δισδιάστατος πίνακας:

Παρατηρούμε ότι ο πίνακας αποτελείται από 6 γραμμές και 5 στήλες, γι αυτό και λέμε ότι ο πίνακας έχει διαστάσεις 6 χ 5. Στην περίπτωση που ο αριθμός των γραμμών και στηλών ενός πίνακα είναι ίδιος, τότε ο πίνακας λέγεται τετραγωνικός.
Σε έναν αλγόριθμο ή σε ένα πρόγραμμα για να συμβολίσουμε έναν δισδιάστατο πίνακα του δίνουμε ένα (επιτρεπτό) όνομα και μέσα σε αγκύλη τον αριθμό των γραμμών και των στηλών του, χωρισμένους με κόμμα. Έτσι, ο παραπάνω πίνακας θα μπορούσε να ονομαστεί table[6,5].
Για να διαβάσει τον παραπάνω πίνακα ένας αλγόριθμος ή ένα πρόγραμμα(δηλ. για να τον «γεμίσει» με δεδομένα) χρησιμοποιεί την επαναληπτική δομή ΓΙΑ…ΑΠΟ…ΜΕΧΡΙ. Έτσι έχω:

```
ΓΙΑ ι ΑΠΟ 1 ΜΕΧΡΙ 6
        ΓΙΑ j ΑΠΟ 1 ΜΕΧΡΙ 5
                ΔΙΑΒΑΣΕ table[ι, j]
        ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
```

όπου τα ι, j είναι οι λεγόμενοι «δείκτες», δηλ. μας δείχνουν σε ποια θέση (γραμμή και στήλη αντίστοιχα) του πίνακα βρισκόμαστε.
Έστω ότι γεμίζω τον πίνακα με τους παρακάτω αριθμούς:

1	2	3	4	5
10	9	8	7	6
11	12	13	14	15
20	19	18	17	16
21	22	23	24	25
30	29	28	27	26

οπότε για ι=1 και j=1 έχω table[1,1]=1, για ι=1 και j=2 έχω table[1,2]=2 κ.ο.κ.

ΠΟΤΕ ΠΡΕΠΕΙ ΝΑ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΠΙΝΑΚΕΣ

Η χρήση πινάκων είναι ένας εύκολος τρόπος για τη διαχείριση πολλών δεδομένων ίδιου τύπου, αλλά πολλές φορές η χρήση τους είναι περιττή. Η απόφαση για τη χρήση ή μη πίνακα είναι θέμα εμπειρίας στον προγραμματισμό. Γενικά, αν τα δεδομένα που εισάγονται στο πρόγραμμα πρέπει να διατηρούνται στη μνήμη μέχρι το τέλος της εκτέλεσης του προγράμματος τότε η χρήση πινάκων βοηθάει ή είναι απαραίτητη για την επίλυση του προβλήματος.

Δυο μειονεκτήματα από την χρήση πινάκων είναι τα εξής:

1. **οι πίνακες απαιτούν πολλή μνήμη, η οποία δεσμεύεται από την αρχή του προγράμματος.**
2. **οι πίνακες περιορίζουν τις δυνατότητες του προγράμματος, καθώς είναι στατικές δομές και το μέγεθος τους πρέπει να καθορίζεται στην αρχή του προγράμματος και παραμένει σταθερό καθ' όλη τη διάρκεια του προγράμματος.**

ΤΥΠΙΚΕΣ ΕΠΕΞΕΡΓΑΣΙΕΣ ΠΙΝΑΚΩΝ

Είναι οι εξής:

1. υπολογισμός αθροισμάτων στοιχείων του πίνακα
2. εύρεση μέγιστου ή ελάχιστου στοιχείου
3. ταξινόμηση των στοιχείων του (ταξινόμηση ευθείας ανταλλαγής)
4. αναζήτηση στοιχείου του πίνακα (σειριακή και δυαδική)
5. συγχώνευση δυο πινάκων