*Στο παρόν βιβλίο παρουσιάζονται οι βασικές έννοιες και θεματικές ενότητες σχετικά με τους Ηλεκτρονικούς Υπολογιστές (Η/Υ) και την επιστήμη της πληροφορικής. Η παρουσίαση των ενοτήτων, θεωρητική κατά βάσιν, είναι απλή και χωρίς να απαιτεί προηγούμενες γνώσεις, ωστόσο κάποια θέματα αναλύονται σε μεγαλύτερο βάθος*

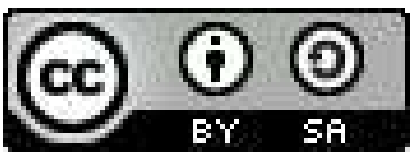
*Τα κύρια θέματα που παρουσιάζονται είναι : η αρχιτεκτονική και τα βασικά δομικά στοιχεία (hardware) του Η/Υ, η οργάνωση και η λειτουργία τους, το Λειτουργικό Σύστημα και οι τεχνικές που χρησιμοποιούνται κατά την εκτέλεση των εργασιών του, τρόποι οργάνωσης δεδομένων, οι πιο γνωστές Δομές Δεδομένων καθώς και τρόποι οργάνωσης αρχείων.*

*Το αρχικό υλικό, που είχε χρησιμοποιηθεί ως σημειώσεις διδασκαλίας, έχει επικαιροποιηθεί, σύμφωνα με τις νεώτερες εξελίξεις του κλάδου των τεχνολογιών πληροφορικής, που ωστόσο είναι συνεχείς τόσο σε έκταση (πεδίο εφαρμογής) όσο και σε βάθος.*

*Νικ.Κοντομίχος*
*Ηλεκτρολόγος Μηχανικός Ε.Μ.Π.*

***Linkedin, facebook : nikolaos kontomichos***

***Αναφορά Δημιουργού – Παρόμοια Διανομή 4.0 (CC-BY-SA)***



**ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ** ……..σελ. 1
1.1. Βασικές έννοιες
1.2. Εξέλιξη των Η/Υ.
1.3. Υλικό και λογισμικό του υπολογιστή
1.4. Εφαρμογές των Η/Υ
1.5. Το ανθρώπινο δυναμικό
**ΚΕΦΑΛΑΙΟ 2. Ο Η/Υ – ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ – ΔΙΚΤΥΑ** …… σελ. 6
2.1 Δομή του υπολογιστή
2.2 Η Κεντρική Μονάδα Επεξεργασίας (Κ.Μ.Ε.)
2.3 Κεντρική Μνήμη (ΚΜ)
2.4 Περιφερειακές Μονάδες
2.4.1 Μονάδες βοηθητικής μνήμης
2.4.2 Μονάδες εισόδου εξόδου
2.4.3 Άλλες περιφερειακές Μονάδες
2.5 Υπολογιστικά συστήματα και τρόποι επεξεργασίας.
2.5.1 Υπολογιστικά συστήματα
2.5.2 Τοπικά Δίκτυα Υπολογιστών (Local Area Networks : LANs) και τοπολογίες
2.5.3 Το Διαδίκτυο (Internet)
2.5.4 Τρόποι επεξεργασίας δεδομένων
**ΚΕΦΑΛΑΙΟ 3. ΑΡΙΘΜΗΤΙΚΑ ΣΥΣΤΗΜΑΤΑ – ΚΩΔΙΚΕΣ – ΣΤΟΙΧΕΙΑ ΑΛΓΕΒΡΑΣ BOOLE**…..σελ. 16
3.1 Αριθμητικά συστήματα
3.2 Το δυαδικό σύστημα. Το οκταδικό και το δεκαεξαδικό σύστημα
3.3 Παράσταση δυαδικών αριθμών στον ΗΥ
3.4 Κωδικοποίηση
3.5 Στοιχεία άλγεβρας Boole
**ΚΕΦΑΛΑΙΟ 4. ΟΡΓΑΝΩΣΗ ΚΑΙ ΛΕΙΤΟΥΡΓΙΑ ΤΗΣ ΚΜΕ** ……. σελ. 23
4.1 Μνήμη υπολογιστή, διεύθυνση, θέσεις μνήμης, λέξη, καταχωρητής
4.2 Εντολές και προγράμματα. Γλώσσα μηχανής και συμβολική γλώσσα
4.3 Κατηγορίες εντολών
4.4 Η μορφή της εντολής
4.5 Μέθοδοι προσδιορισμού διεύθυνσης μνήμης
4.6 Τρόπος λειτουργίας της ΚΜΕ και καταχωρητές
4.7 Συμβολική Γλώσσα (Assembly)
4.8 Μικροεπεξεργαστές και αρχιτεκτονικές σύγχρονων μικροεπεξεργαστών
**ΚΕΦΑΛΑΙΟ 5 ΟΡΓΑΝΩΣΗ ΤΗΣ ΜΝΗΜΗΣ** ……..……. σελ. 30
5.1 Η μνήμη του υπολογιστή. Βοηθητική μνήμη.
5.2 Γρήγορη Μνήμη (cache memory)
5.3 Η συσχετιστική μνήμη (associative memory)
5.4 Ιδεατή μνήμη (virtual memory)
5.4.1 Απεικόνιση διευθύνσεων
**ΚΕΦΑΛΑΙΟ 6. ΤΟ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ** …… σελ. 36
6.1 Γενική παρουσίαση του λειτουργικού συστήματος
6.1.1 Εισαγωγή
6.1.2 Πυρήνας και φλοιός του λειτουργικού συστήματος
6.1.3 Προγράμματα του λειτουργικού συστήματος
6.1.4 Καταμερισμός χρόνου (time sharing)
6.1.5 Συστήματα multi user και multi tasking
6.2 Διαχείριση της ΚΜΕ.
6.2.1 Η έννοια της διεργασίας (process)
6.2.2 Χρονοδρομολόγηση της ΚΜΕ
6.2.3 Αλγόριθμοι χρονοδρομολόγησης

6.3 Διαχείριση μνήμης
6.3.1 Αμοιβαία ανταλλαγή (swapping)
6.3.2 Overlays και ιδεατή μνήμη
6.3.3 Σελιδοποίηση (paging)
6.3.4. Μέγεθος σελίδας και κατακερματισμός (fragmentation)
6.3.5 Κατάτμηση (segmentation)
6.4 Διαχείριση αρχείων
6.4.1 Αρχεία και μέσα αποθήκευσης
6.4.2 Κατάλογοι Αρχείων
6.4.3 Ιεραρχική δομή αρχείων
6.4.4 Εντολές αρχείων
**ΚΕΦΑΛΑΙΟ 7. ΑΛΓΟΡΙΘΜΟΙ – ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ – ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ** ….. 50
7.1 Προγράμματα και προγραμματισμός
7.1.1 Τεχνικές προγραμματισμού
7.2 Αλγόριθμοι και λογικά διαγράμματα
7.3 Γλώσσες προγραμματισμού
7.3.1 Κατηγορίες γλωσσών προγραμματισμού
**ΚΕΦΑΛΑΙΟ 8. .ΔΕΔΟΜΕΝΑ ΚΑΙ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ** …… 59
8.1 Η έννοια των δεδομένων (data) και η παράστασή τους στον Η/Υ
8.2 Μελέτη των δεδομένων. Τύποι δεδομένων
8.2.1 Απλοί ή βαθμωτοί τύποι
8.2.2 Σύνθετοι τύποι δεδομένων ή Δομές Δεδομένων
8.2.3 Μελέτη της μορφής μιας δομής δεδομένων
8.2.4 Βασικές λειτουργίες σε μία δομή δεδομένων.
8.3 Δομές δεδομένων και αλγόριθμοι
8.4 Πίνακες
8.4.1 Γενικά
8.4.2 Ειδικές μορφές πινάκων
8.4.3 Πίνακες συμβολοσειρών
8.5 Γραμμικές λίστες
8.5.1. Γενικά
8.5.2 Σειριακές γραμμικές λίστες
8.5.2.1 Στοίβα
8.5.2.2 Ουρά
8.5.2.3 Ουρά με δύο άκρα
8.5.3 Συνδεδεμένες γραμμικές λίστες
8.6 Ταξινόμηση πινάκων
**ΚΕΦΑΛΑΙΟ 9. ΟΡΓΑΝΩΣΗ ΑΡΧΕΙΩΝ** ……... 71
9.1 Αρχεία και εγγραφές
9.2 Είδη αρχείων
9.3. Μέθοδοι οργάνωσης και αναζήτησης αρχείων
9.3.1 Σειριακή (sequential) οργάνωση
9.3.2 Σειριακή οργάνωση με δείκτη (indexed sequential)
9.3.3 Απ'ευθείας (direct ή random) οργάνωση
9.3.4 Αναζήτηση με βάση δευτερεύοντα κλειδιά
**ΚΕΦΑΛΑΙΟ 10. ΕΦΑΡΜΟΓΕΣ ΤΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ – ΕΙΔΙΚΑ ΘΕΜΑΤΑ** …… 78
10.1 Εφαρμογές των Η/Υ
10.2 Η πληροφορική στην εκπαίδευση
10.3 Ασφάλεια υπολογιστικών συστημάτων

# ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

## 1.1. Βασικές έννοιες

Πληροφορική είναι η επιστήμη που ασχολείται με την παράσταση, αποθήκευση, επεξεργασία και παρουσίαση πληροφοριών.
Βασικό "εργαλείο" στην επιστήμη της πληροφορικής είναι ο ηλεκτρονικός υπολογιστής (Η/Υ). Ο Η/Υ (υπολογιστής ή computer) είναι μία αυτοματοποιημένη ηλεκτρονική μηχανή που δέχεται δεδομένα (πληροφορίες) τα επεξεργάζεται και δίνει στην έξοδο κάποια αποτελέσματα (πληροφορίες).
Ο Η/Υ μπορεί να εκτελεί υπολογισμούς με μεγάλη ταχύτητα και ακρίβεια (η ταχύτητά του είναι κατά πολύ μεγαλύτερη από εκείνη του ανθρώπινου εγκεφάλου) αλλά δεν μπορεί να "σκεφτεί" με δημιουργικό τρόπο όπως ο ανθρώπινος εγκέφαλος. Λειτουργεί πάντα αιτιοκρατικά, δηλ. με νόμους και κανόνες που ο άνθρωπος καθορίζει.
Η πληροφορική και οι Η/Υ παίζουν σήμερα έναν πολύ σπουδαίο ρόλο στη ζωή, ώστε να λέμε ότι έχει ήδη συντελεστεί μια νέα επανάσταση (η επανάσταση της πληροφορικής) που η σημασία της δεν είναι μικρότερη από τη βιομηχανική επανάσταση, και μάλιστα με την πάροδο του χρόνου γίνεται διαρκώς μεγαλύτερη.

## 1.2. Εξέλιξη των Η/Υ.

Η ιστορία των Η/Υ ανάγεται στις πρώτες προσπάθειες του ανθρώπου να κάνει υπολογισμούς με κάποιον αυτοματοποιημένο τρόπο. Ως πρώτη υπολογιστική συσκευή θεωρείται ο άβακας, που αποτελείται από σφαιρίδια τα οποία μετακινούνται κατά μήκος των χορδών πάνω στις οποίες στηρίζονται. Ανάλογα με τη θέση όπου τοποθετούμε τα σφαιρίδια παριστάνουμε κάποιον αριθμό. Ο άβακας χρησιμοποιείτο ήδη απο το 3.000 π.Χ. απο τους Κινέζους και τους Ιάπωνες.
Τον 17ο αιώνα ο Σκώτος μαθηματικός John Napier επινοεί τους λογαρίθμους και απλοποιεί τις πράξεις του πολλαπλασιασμού και της διαίρεσης. Παράλληλα κατασκευάζει και μια απλή υπολογιστική συσκευή (ράβδοι του Napier). Το 1642 ο Γάλλος μαθηματικός Pascal κατασκευάζει την πρώτη αριθμητική μηχανή που εκτελεί προσθέσεις και αφαιρέσεις χρησιμοποιώντας οδοντωτούς τροχούς και μοχλούς. Ο Γερμανός μαθηματικός Leibniz το 1673 βελτιώνει τη μηχανή του Pascal ώστε να εκτελεί πολλαπλασιασμούς και διαιρέσεις. Ακολούθησαν διάφορα βήματα για την εξέλιξη των υπολογιστικών αυτών συσκευών. Ιδιαίτερο ενδιαφέρον παρουσιάζει η υφαντουργική μηχανή του Jacquard στις αρχές του 18ου αιώνα στη Γαλλία, που με διάτρητες κάρτες έλεγχε αυτόματα την ύφανση πολύπλοκων σχεδίων.

Στις αρχές του 19ου αιώνα ο Αγγλος Charles Babbage, καθηγητής στο πανεπιστήμιο του Cambridge, συλλαμβάνει την ιδέα της κατασκευής μίας αυτόματης υπολογιστικής μηχανής. Επινοεί τη Διαφορική Μηχανή και στη συνέχεια την Αναλυτική Μηχανή, που θεωρείται πρόγονος των σημερινών Η/Υ. Περιείχε στοιχεία μνήμης, μονάδα ελέγχου, μονάδα αριθμητικών πράξεων και μονάδες εισόδου εξόδου. Η κατασκευή και των δύο αυτών μηχανών δεν ολοκληρώθηκε, κυρίως διότι δεν υπήρχε την εποχή εκείνη η κατάλληλη τεχνολογία για την κατασκευή των απαιτούμενων εξαρτημάτων. Προς το τέλος του 19ου αιώνα ο Herman Hollerith στις Η.Π.Α. κατασκευάζει την πρώτη ηλεκτρομηχανολογική υπολογιστική μηχανή που λειτουργούσε με διάτρητες κάρτες. Εν τω μεταξύ το 1854 ο ‘Αγγλος μαθηματικός G. Boole επινόησε την Αλγεβρα των δυαδικών αριθμών (Αλγεβρα Boole), η οποία βρήκε αργότερα εφαρμογή στα ψηφιακά κυκλώματα που χρησιμοποιούνται στους Η/Υ.

Ο πρώτος υπολογιστής "εγγεγραμμένων προγράμματος" ολοκληρώνεται το 1944 από τον Η. Aiken του Πανεπιστημίου του Harvard σε συνεργασία με την IBM, και ονομάστηκε MARK I. Λειτουργούσε με ηλεκτρονόμους (ρελαί) και διάτρητες κάρτες. ‘Ηταν ένα μηχάνημα μεγάλο σε όγκο και βάρος (ζύγιζε περίπου 5 τόνους), πολύπλοκο και πολύ αργό σε σχέση με τους μεταγενέστερους Η/Υ. Πρώτος ψηφιακός υπολογιστής είναι ο ENIAC που ολοκληρώθηκε το 1945 στο πανεπιστήμιο της Pensylvania. ‘Ηταν 100 φορές ταχύτερος απο τον MARK I. Στη συνέχεια η εξέλιξη που σημειώνεται στους Η/Υ είναι ραγδαία και οι νέοι Η/Υ που παρουσιάζονται πολλοί.

Το 1951 κατασκευάζεται ο UNIVAC I, ο πρώτος Η/Υ για εμπορικούς σκοπούς. Το 1952 τίθεται σε λειτουργία ο IAS, ένας Η/Υ βασισμένος πάνω στις ιδέες του John von Neuman του πανεπιστημίου του Princeton, για την λογική σχεδίαση υπολογιστικών μηχανών.

Εν τω μεταξύ η εξέλιξη των Η/Υ χωρίζεται σε "γενεές". Οι Η/Υ όσο εξελίσσονται, μικραίνουν δραματικά σε μέγεθος, ενώ η ταχύτητα και η ισχύς τους αυξάνονται εντυπωσιακά, χάρη στην πρόοδο της ηλεκτρονικής τεχνολογίας.

Στην πρώτη γενεά Η/Υ χρησιμοποιούνται ηλεκτρονικές λυχνίες. Οι υπολογιστές της πρώτης γενεάς έχουν μεγάλο όγκο και μικρή ταχύτητα υπολογισμών. Στη δεύτερη γενεά, που αρχίζει το 1959 χρησιμοποιούνται ημιαγωγοί. Στην τρίτη γενεά που αρχίζει από το 1965 χρησιμοποιούνται ολοκληρωμένα κυκλώματα. Οι διαστάσεις πλέον μειώνονται πάρα πολύ ενώ η ταχύτητα υπολογισμών φθάνει στην τάξη μεγέθους του νανοδευτερολέπτου (1 nsec=$10^{-9}$ sec). Απο το 1970 έχομε την τέταρτη γενεά υπολογιστών με πολλές τεχνολογικές εξελίξεις.

Από το 1980 παρουσιάζονται οι προσωπικοί υπολογιστές (PC: personal computers), που λόγω του χαμηλού κόστους διαδόθηκαν ευρέως, ενώ παράλληλα άρχισε να αναδεικνύεται όλο και σημαντικότερος ο ρόλος του λογισμικού (software), που στην αρχή της εξέλιξης των Η/Υ ήταν ανύπαρκτος. Παράλληλα με τους επιτραπέζιους προσωπικούς υπολογιστές (desktop personal computers) εμφανίζονται και διαδίδονται ταχύτατα και οι φορητοί

υπολογιστές (laptop computers) που ενσωματώνουν πληκτρολόγιο, οθόνη και ποντίκι σε μία φορητή συσκευή, με δυνατότητα παροχής ηλεκτρικού ρεύματος από επαναφορτιζόμενες μπαταρίες. Από το 2000 χρησιμοποιούνται ευρέως και νέες φορητές συσκευές όπως οι υπολογιστές-ταμπλέτες (tablets) και τα έξυπνα τηλέφωνα (smartphones), που και αυτά είναι μικροί προσωπικοί υπολογιστές.

Η εξέλιξη των Η/Υ έφερε νέες τεχνικές και δυνατότητες, όπως.

- Πολυπρογραμματισμός (multiprogramming)
- Πολυεπεξεργασία (multiprocessing)
- Συστήματα πραγματικού χρόνου (real time)
- Συστήματα on-line
- Μεγάλης κλίμακας υπολογιστές (mainframes) και Υπερυπολογιστές (supercomputers)
- Πολυμέσα (multimedia)
- Τεχνητή νοημοσύνη (artificial intelligence)

ενώ η ανάπτυξη του Διαδικτύου (Internet) και η δυνατότητα σύνδεσης των Η/Υ σε αυτό, που συνοδεύτηκαν με την ανάπτυξη προγραμμάτων περιήγησης στο Διαδίκτυο (μηχανές αναζήτησης), έφερε νέα επανάσταση στις δυνατότητες και εφαρμογές του Η/Υ.

Τέλος, σήμερα πολλές μικρές και μεγαλύτερες συσκευές περιέχουν μικρούς Η/Υ.

### 1.3. Υλικό και λογισμικό του υπολογιστή

Οποιοδήποτε σύστημα Η/Υ αποτελείται από δύο μέρη που διακρίνονται το ένα από το άλλο αλλά συνεργάζονται στενά κατά τη λειτουργία του : Το υλικό (hardware) και το λογισμικό (software).

Υλικό είναι το ηλεκτρονικό, ηλεκτρικό και μηχανικό μέρος του Η/Υ ενώ Λογισμικό είναι τα προγράμματα. Πρόγραμμα είναι μία ακολουθία εντολών, γραμμένες σε κατάλληλη γλώσσα, κατανοητή από τον Η/Υ.

Ο υπολογιστής μπορεί να εκτελέσει πολλές και διαφορετικές υπολογιστικές εργασίες (Η/Υ γενικού σκοπού). Εκείνο που χρειάζεται σε κάθε περίπτωση είναι το κατάλληλο πρόγραμμα.

Το λογισμικό χωρίζεται σε δύο μεγάλες κατηγορίες:

α) Προγράμματα συστήματος (System Software). Είναι εκείνα τα προγράμματα που είναι απαραίτητα για την αξιοποίηση και λειτουργία του υλικού. Το σημαντικότερο πρόγραμμα από αυτά είναι το Λειτουργικό Σύστημα (θα αναλυθεί στο Κεφάλαιο 6. Υπάρχουν επίσης άλλα βοηθητικά προγράμματα (utilities) που βελτιώνουν την απόδοση του λειτουργικού συστήματος και αξιοποιούν καλλίτερα τις δυνατότητες του.

β) Προγράμματα εφαρμογών (Application Software). Προγράμματα που καλύπτουν όλο το φάσμα χρήσεων και εφαρμογών του υπολογιστή. Δημιουργούνται από εταιρείες παραγωγής λογισμικού ή και από μεμονωμένους χρήστες ή κοινότητες χρηστών.

Διακρίνουμε το κλειστό ή εμπορικό λογισμικό (propritary software) που μπορεί κανείς να αγοράσει, αλλά και το λογισμικό ανοιχτού κώδικα (open source) που μπορεί κανείς να κατεβάσει δωρεάν από το Internet.
Τόσο τα σύγχρονα λειτουργικά συστήματα όσο και το λογισμικό εφαρμογών, χρησιμοποιούν τεχνικές διασύνδεσης γραφικών με το χρήστη (graphical user inerface - GUI), ώστε ο χρήστης να έχει στη διάθεσή του στην οθόνη εικονίδια, κουμπιά, ράβδους, πλαίσια διαφόρων σχημάτων και μεγεθών,με τελικό σκοπό η εκτέλεση εργασιών να γίνεται πιο εύκολη και φιλική στο χρήστη (user friendly interface).

### 1.4. Εφαρμογές των Η/Υ

Ο Η/Υ ξεκίνησε σαν μία μηχανή που έκανε βασικά υπολογισμούς. Με την πάροδο του χρόνου το εύρος των εφαρμογών του βαίνει διαρκώς αυξανόμενο. Ήδη από τη δεκαετία του 1970 οι Η/Υ χρησιμοποιούνταν σε περιοχές εφαρμογών όπως:

- Επίλυση υπολογιστικών προβλημάτων για επιστημονικές εφαρμογές
- Προγράμματα στατιστικής επεξεργασίας
- Εμπορικές εφαρμογές (για εταιρείες, επιχειρήσεις, οργανισμούς κ.λπ.) π.χ.: μισθοδοσία, λογιστήριο, παρακολούθηση κίνησης αποθήκης, οικονομικός προγραμματισμός κ.λπ.
- Τραπεζικές εφαρμογές π.χ. ενημέρωση καταθέσεων/αναλήψεων, δάνεια, διεκπεραίωση επιταγών κ.λπ.
- Σε υπουργεία και οργανισμούς κοινής ωφέλειας, π.χ. καταχώρηση-έλεγχος φορολογικών δηλώσεων, έκδοση λογαριασμών ηλεκτρικού ρεύματος, τηλεφωνίας, ύδρευσης κ.λπ.
- Σε αεροπορικές, σιδηροδρομικές, ναυτιλιακές εταιρείες π.χ. κράτηση θέσεων, προγραμματισμός δρομολογίων κ.λπ.
- Στη βιομηχανία σε συστήματα αυτόματης παρακολούθησης και ελέγχου μηχανημάτων, λειτουργία εργαλειομηχανών, βιομηχανικών ρομπότ κ.λπ.
- Στην ιατρική, για ανάλυση δεδομένων και μετρήσεων κ.λ.π.
- Σχεδιαστικά προγράμματα για σχεδίαση κτιρίων, δρόμων, γεφυρών, αεροδρομίων, αεροπλάνων, αυτοκινήτων, μηχανών, δικτύων κ.λπ.
- Προγράμματα και εφαρμογές αυτοματισμού γραφείου, όπως λογιστικά φύλλα (spreadsheets) και επεξεργαστές κειμένου (word processors).
- Παιχνίδια (games).

Όπως αναφέρθηκε ήδη, η δυνατότητα σύνδεσης στο Διαδικτύου, σε συνδυασμό με την ανάπτυξη προγραμμάτων και εφαρμογών λογισμικού, έφερε τις τελευταίες δεκαετίες πρακτικά απεριόριστες δυνατότητες και εφαρμογές για τους Η/Υ, σχεδόν σε κάθε τομέα της ζωής σήμερα. Πολλές από τις εφαρμογές αυτές παρέχονται δωρεάν (μπορεί κάποιος να τις

κατεβάσει ελεύθερα από το Διαδίκτυο). Μερικά χαρακτηριστικά παραδείγματα εφαρμογών που χρησιμοποιούνται στην καθημερινή ζωή είναι :

- Ηλεκτρονικό ταχυδρομείο (email) καθώς και εφαρμογές (πλατφόρμες) ανταλλαγής μηνυμάτων και αρχείων και φωνητικών κλήσεων μέσω Διαδικτύου
- Εφαρμογές Κοινωνικής Δικτύωσης. Τα κοινωνικά δίκτυα (social media) έχουν γνωρίσει εκρηκτική ανάπτυξη.
- Εφαρμογές για δημοσίευση αρχείων εικόνας, video, ήχου και διαμοιρασμού περιεχομένου
- Εφαρμογές για πραγματοποίηση τηλεδιάσκεψης μέσω Διαδικτύου.
- Ηλεκτρονικό εμπόριο (e-commerce) και ηλεκτρονικό marketing (digital marketing) για επιχειρήσεις.

… και άλλες πρακτικά απεριόριστες εφαρμογές.

### 1.5. Το ανθρώπινο δυναμικό

Οι κυριότερες θέσεις εργασίας που σχετίζονται με ένα υπολογιστικό σύστημα και τις εφαρμογές που αναπτύσσονται σ' αυτό είναι:

Ο αναλυτής (analyst) που κάνει τη μελέτη και ανάλυση ενός προβλήματος και δίνει τις γενικές κατευθύνσεις για τη λύση του.

Ο προγραμματιστής (programmer) υλοποιεί τη λύση που δίνει ο αναλυτής σε πρόγραμμα. Κάνει επίσης τις δοκιμές που απαιτούνται στο πρόγραμμα και την τεκμηρίωσή του.

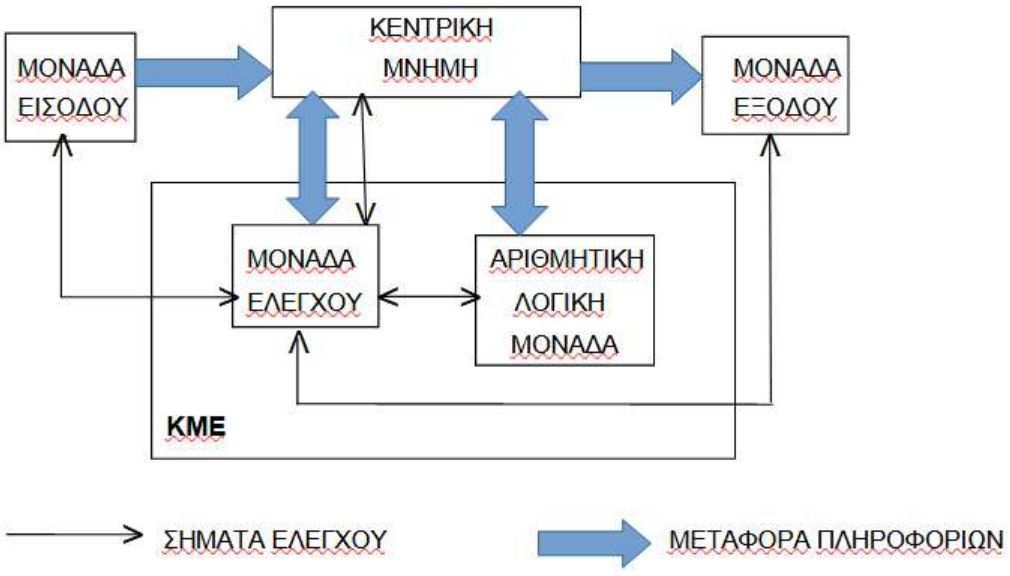
Ο χειριστής/χρήστης (user) εκτελεί το πρόγραμμα, δίνει τα δεδομένα και παίρνει τα αποτελέσματα.

Ο διαχειριστής (administrator) είναι υπεύθυνος για τη συνολική λειτουργία ενός μεγάλου υπολογιστικού συστήματος ή και δικτύου υπολογιστών, όπου "τρέχουν" διάφορες εφαρμογές. Βέβαια οι τεχνολογικές εξελίξεις δημιουργούν διαρκώς νέες ανάγκες και εξειδικευμένες θέσεις εργασίας, για παράδειγμα Ειδικός Κυβενοασφάλειας, Ειδικός/Αναλυτής Δεδομένων (Data Scientist/Analyst), Μηχανικός Τεχνητής Νοημοσύνης κ.α.

# ΚΕΦΑΛΑΙΟ 2. Ο Η/Υ – ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ – ΔΙΚΤΥΑ

## 2.1. Δομή του υπολογιστή

‘Ενας Η/Υ αποτελείται από τις εξής βασικές λειτουργικές μονάδες.
α) Κεντρική Μονάδα Επεξεργασίας (Κ.Μ.Ε. ή CPU: Central Processing Unit).
β) Κεντρική Μνήμη (Κ.Μ. ή Main Memory)
γ) Περιφερειακές Μονάδες (Peripheral Units)
Η Κ.Μ.Ε. αποτελείται από την Αριθμητική/Λογική Μονάδα (ΑΛΜ ή ALU: Arithmetic/Logical Unit) και την Μονάδα Ελέγχου (Μ.Ε. ή Control Unit).
Περιφερειακές μονάδες είναι οι Μονάδες Εισόδου/Εξόδου (Μονάδες I/O : Input/Output Units) οι μονάδες βοηθητικής μνήμης καθώς και μερικές άλλες.
Η λειτουργία ενός υπολογιστή ανεξάρτητα από το είδος της εργασίας, ακολουθεί το εξής σχήμα: Εισαγωγή δεδομένων - επεξεργασία - έξοδος αποτελεσμάτων. Πιο αναλυτικά, από κάποια μονάδα εισόδου η βοηθητικής μνήμης μεταφέρεται στην Κεντρική Μνήμη κάποιο πρόγραμμα και κάποια δεδομένα. Η Κ.Μ.Ε. "εκτελεί" το πρόγραμμα και πιο συγκεκριμένα η Μονάδα Ελέγχου λαμβάνει (υπό μορφή ηλεκτρικών σημάτων) τις εντολές του προγράμματος από την Κεντρική Μνήμη και τις μεταφέρει κατάλληλα κωδικοποιημένες στην Αριθμητική/Λογική Μονάδα. Από την επεξεργασία των δεδομένων με βάση τις εντολές του προγράμματος προκύπτουν τα αποτελέσματα που μεταφέρονται είτε απ' ευθείας σε κάποια μονάδα εξόδου, είτε στην Κεντρική Μνήμη και από εκεί σε κάποια μονάδα εξόδου.
Παρακάτω θα δούμε αναλυτικότερα τις μονάδες ενός Η/Υ




**Σχήμα 1. Οι μονάδες του Η/Υ**

## 2.2. Η Κεντρική Μονάδα Επεξεργασίας (Κ.Μ.Ε.)

Αναφέρεται πιο απλά ως επεξεργαστής (processor). Τα πιο σημαντικά χαρακτηριστικά ενός επεξεργαστή είναι η αρχιτεκτονική του και η ταχύτητα του που μετριέται σε εντολές ανά δευτερόλεπτο (i.p.s. Instructions per second) Σε πιο απλά υπολογιστικά συστήματα (όπως στους προσωπικούς υπολογιστές) ο επεξεργαστής αποτελείται από ένα μόνο ολοκληρωμένο κύκλωμα (μικροεπεξεργαστής ή microprocessor).

Οι υπομονάδες της ΚΜΕ

α) Η αριθμητική/λογική μονάδα (ΑΛΜ) πραγματοποιεί αριθμητικές πράξεις, λογικές πράξεις (ΚΑΙ, Η, Αντιστροφή, ολίσθηση) συγκρίσεις και βάσει αυτών κάνει υπολογισμούς. Οι κωδικοποιημένες παραστάσεις δεδομένων εισέρχονται στην ΑΛΜ από την ΚΜ. Η μονάδα ελέγχου (ΜΕ) παρέχει στην ΑΛΜ τα απαιτούμενα σήματα ελέγχου σε προκαθορισμένες χρονικές στιγμές ώστε αυτή να εκτελέσει τις προκαθορισμένες αριθμητικές και λογικές πράξεις

β) Η μονάδα ελέγχου (ΜΕ) ελέγχει και συντονίζει τις λειτουργίες των υπολοίπων μονάδων του Η/Υ. Η ΜΕ αποκωδικοποιεί τις εντολές του προγράμματος και εκπέμπει σε καθορισμένες χρονικές στιγμές σήματα ελέγχου προς την ΚΜ την ΑΛΜ και τις περιφερειακές μονάδες Επίσης συγχρονίζει τις λειτουργίες του υπολογιστή με βάση τους παλμούς του ρολογιού (clock).

## 2.3 Η Κεντρική Μνήμη (Κ.Μ.)

Είναι ο χώρος αποθήκευσης δεδομένων και προγραμμάτων.

Παραστατικά μπορούμε να την αντιληφθούμε σαν ένα σύνολο θέσεων καθεμία από τις οποίες έχει μία διεύθυνση και μπορεί να δεχτεί (να αποθηκεύσει) μία τιμή.

Η παράσταση (κωδικοποίηση) των δεδομένων γίνεται στο δυαδικό σύστημα που έχει δύο μόνο τιμές, το 0 και το 1. Η στοιχειώδης μονάδα οργάνωσης της ΚΜ είναι το Byte. 1 Byte αποτελείται από οκτώ Bits (Binary Digit = δυαδικό ψηφίο, δηλαδή ψηφίο του δυαδικού συστήματος 0 ή 1).

Κύρια χαρακτηριστικά μνήμης είναι η χωρητικότητα και η ταχύτητα προσπέλασης (σε μία διεύθυνση). Η χωρητικότητα μετριέται σε πολλαπλάσια του Byte δηλ. KBytes (1 KByte=$2^{10}$ Bytes=1024 Bytes), MBytes (1 MByte=$2^{10}$ KBytes= $2^{20}$ Bytes= 1048576 Bytes), Gbytes (1 GByte=$2^{10}$ MBytes), Tbytes κλπ.

Η ταχύτητα προσπέλασης της μνήμης μετριέται σε υποπολλαπλάσια του δευτερολέπτου (1ms=$10^{-3}$ sec, 1μsec=$10^{-6}$ sec, 1nsec=$10^{-9}$ sec).

Στην ΚΜ καταχωρούνται δεδομένα, προγράμματα και αποτελέσματα. Η ΚΜ ανταλλάσσει πληροφορίες με την ΑΛΜ και δέχεται σήματα από την ΜΕ για μεταφορά και αποθήκευση

δεδομένων. Επίσης εξυπηρετεί τη ροή πληροφοριών μεταξύ ΚΜΕ και περιφερειακών μονάδων.

Είδη μνήμης

RAM (Random Access Memory = μνήμη τυχαίας προσπέλασης)

Είναι η μνήμη που έχει στη διάθεσή του ο χρήστης όπου μπορεί να αποθηκεύσει προγράμματα δεδομένα κ.λ.π. και στη συνέχεια να τα ανακαλέσει. Όταν κλείνουμε τον Η/Υ τα περιεχόμενα της χάνονται.

ROM (Read Only Memory). Από τη μνήμη αυτή ο χρήστης μπορεί να διαβάσει (δηλαδή να ανακαλέσει) δεδομένα αλλά δεν μπορεί να αποθηκεύσει σε αυτήν δεδομένα (δηλαδή να αλλάξει το περιεχόμενο της). Όταν κλείσουμε τον Η/Υ τα περιεχόμενα της δεν χάνονται.

PROM (Programmable ROM). Είναι μνήμη που μπορεί να "προγραμματιστεί" δηλαδή να καταχωρήσουμε σε αυτήν στοιχεία μόνο μία φορά και στη συνέχεια συμπεριφέρεται σαν ROM.

EPROM (Erasable PROM). Είναι σαν την PROM αλλά με ειδική διαδικασία, με υπεριώδεις ακτίνες, το περιεχόμενό της μπορεί να σβηστεί και στη συνέχεια να ξαναπρογραμματιστεί.

EEPROM (Electrically EPROM). Είναι σαν την EPROM, αλλά το περιεχόμενό της σβήνεται με ηλεκτρικά σήματα.

Η Κεντρική Μνήμη του Η/Υ περιλαμβάνει την RAM και την ROM.

## 2.4. Περιφερειακές Μονάδες

### 2.4.1. Μονάδες βοηθητικής μνήμης

Είναι μαγνητικά κυρίως μέσα αποθήκευσης δεδομένων. Διακρίνονται σε μέσα σταθερής κεφαλής (μαγνητικές ταινίες) και μέσα κινητής κεφαλής (δίσκους) Σε έναν Η/Υ μπορεί να υπάρχουν μέσα και των δύο κατηγοριών.

Πιο διαδεδομένα μέσα βοηθητικής μνήμης είναι οι δίσκοι (discs). Ένας δίσκος αποτελείται από ένα σύνολο από πλάκες. Η ανάγνωση και εγγραφή δεδομένων (πληροφορία) γίνεται και στις δύο επιφάνειες κάθε πλάκας. Κάθε πλευρά της κάθε πλάκας αποτελείται από ομόκεντρους κύκλους (άτρακτοι ή ίχνη : tracks) και κάθε κύκλος χωρίζεται σε ισομεγέθη τμήματα (τομείς : sectors). Κάθε τομέας περιέχει ορισμένο αριθμό bytes.

Τα κυριότερα είδη βοηθητικής μνήμης είναι :

- Η μαγνητική ταινία (magnetic tape) – εξωτερική μονάδα αποθήκευσης, που χρησιμοποιείται πλέον σπάνια
- Ο εύκαμπτος δίσκος (floppy disk – δισκέτα) – εξωτερική μονάδα αποθήκευσης, χρησιμοποιείται πλέον σπάνια
- Ο σκληρός δίσκος (HD - hard disk) - εσωτερική μονάδα αποθήκευσης
- Ο οπτικός δίσκος (CD, DVD, Blu-ray) - εξωτερική μονάδα αποθήκευσης
- Η μνήμη στερεάς κατάστασης (SSD – Solid State Drive) - εσωτερική μονάδα αποθήκευσης

- Η μνήμη ενιαίου σειριακού διαύλου (flash memory with USB–Universal Serial Bus interface)
- εξωτερική μονάδα αποθήκευσης

Κυριότερα χαρακτηριστικά των μέσων βοηθητικής μνήμης είναι η χωρητικότητα (Mbytes/Gbytes/Tbytes), η ταχύτητα προσπέλασης (msec) και το κόστος.

Εκτός από τις αυτόνομες μονάδες αποθήκευσης ενός υπολογιστικού συστήματος, χρησιμοποιείται όλο και περισσότερο η τεχνική αποθήκευσης στο υπολογιστικό "νέφος" (cloud), που σημαίνει οτι τα δεδομένα αποθηκεύονται σε μονάδες αποθήκευσης που βρίσκονται σε υπολογιστικά συστήματα σε άλλο γεωγραφικό σημείο (κυρίως σε Data Centers), με τα οποία ο Η/Υ συνδέεται και επικοινωνεί.

### 2.4.2 Μονάδες Εισόδου/Εξόδου (Ε/Ε) ή Μονάδες Input-Output (I/O)

#### 2.4.2.1 Περιγραφή μονάδων εισόδου/εξόδου

Είναι μονάδες μέσω των οποίων επικοινωνεί ο χρήστης με τον Η/Υ. Από τη μονάδα εισόδου ο χρήστης στέλνει στοιχεία στον Η/Υ (δεδομένα ή πρόγραμμα) και από τη μονάδα εξόδου παίρνει τα αποτελέσματα Στα μεγάλα υπολογιστικά συστήματα οι Μονάδες εισόδου και εξόδου μπορεί να βρίσκονται μακριά από τις κεντρικές μονάδες του υπολογιστή.

Οι μονάδες Ε/Ε μετατρέπουν τα δεδομένα που εισάγει ο χρήστης σε μορφή κατανοητή από τον Η/Υ και αντίστροφα μετατρέπουν τα αποτελέσματα της επεξεργασίας του Η/Υ σε μορφή κατανοητή από τον χρήστη. Θα δούμε στη συνέχεια μία σύντομη περιγραφή των πιο βασικών μονάδων Ε/Ε.

α) <u>Το πληκτρολόγιο (keyboard)</u>

Περιλαμβάνει ένα σύνολο από πλήκτρα καθένα από τα οποία αντιστοιχεί σε κάποιο γράμμα, αριθμό, σημείο στίξης κλπ. Υπάρχουν όμως και πλήκτρα που εκτελούν συγκεκριμμένες λειτουργίες (function keys) ενώ ορισμένα πλήκτρα μπορεί να αλλάζουν λειτουργία σε συνδυασμό με το πάτημα κάποιου άλλου πλήκτρου ταυτόχρονα (Control, Shift, Alt κλπ). Ακόμα, το ίδιο το πλήκτρο μπορεί να επιτελέι διαφορετική λειτουργία ανάλογα με το περιβάλλον που βρισκόμαστε.

Όταν πιέζουμε κάποιο πλήκτρο στην πραγματικότητα κλείνουμε κάποιον διακόπτη και στέλνουμε προς τον υπολογιστή έναν συνδυασμό από bits που είναι ο κώδικας (κατά ASCII ή σύμφωνα με κάποια άλλη κωδικοποίηση) του γράμματος, αριθμού κλπ που αντιστοιχεί στο συγκεκριμμένο πλήκτρο) Το πληκτρολόγιο που συνήθως χρησιμοποιείται λέγεται QWERTY από τα πρώτα γράμματα της πρώτης σειράς,

β) <u>Η οθόνη (screen ή monitor)</u>

Η οθόνη του υπολογιστή μοιάζει με την οθόνη της τηλεόρασης. Υπάρχουν μονόχρωμες και έγχρωμες οθόνες. Σήμερα η πλειοψηφία των οθονών είναι έγχρωμες, τεχνολογίας LCD (Liquid Crystal Display) και LED (Light Emitting Diode). Κυριότερα χαρακτηριστικά μίας

οθόνης είναι το μέγεθος και η ανάλυση, που εκφράζεται σαν πίνακας δύο διαστάσεων από εικονοστοιχεία (pixels = picture elements). Το μέγεθός της, συνήθως σε ίντσες, εκφράζει το μήκος της διαγωνίου της. Για να επικοινωνήσει ο Η/Υ με την οθόνη εξοπλίζεται με ειδική κάρτα οθόνης (interface οθόνης η κάρτα γραφικών). Το πιο σημαντικό χαρακτηριστικό μιάς οθόνης είναι η διακριτότητά της (resolution) που εκφράζει τον αριθμό των εικονοστοιχείων που εμφανίζονται στην οθόνη. Έτσι υπάρχουν οθόνες υψηλής διακριτότητας (high resolution) που είναι κατάλληλες για εφαρμογές με γραφικές παραστάσεις, σχεδιαστικά προγράμματα ή video games, και απλές οθόνες που είναι κατάλληλες κυρίως για απλά κείμενα, σχήματα κλπ. Η διακριτότητα καθορίζεται από την κάρτα γραφικών της οθόνης.

Τα πιο διαδεδομένα συστήματα που χρησιμοποιούνται σήμερα για τη σύνδεση μεταξύ της κάρτας βίντεο και της οθόνης του υπολογιστή είναι:

- Video Graphics Array (VGA) : επιτρέπει 256 χρώματα και την εμφάνιση εικόνων
- Digital Video Interface (DVI): για οθόνες όπως οι LCD, οι LED, οι οθόνες πλάσματος
- High-Definition Multimedia Interface (HDMI) -σύστημα μετάδοσης ψηφιακού ήχου και βίντεο

Στην έγχρωμη οθόνη κάθε σημείο προκύπτει ως σύνθεση τριών βασικών χρωμάτων κόκκινο, πράσινο και μπλέ (RGB – Red-Green-Blue). Οι οθόνες που χρησιμοποιούνται σήμερα είναι κυρίως τεχνολογίας υγρών κρυστάλλων (LCD : Liquid Crystal Display). Οι οθόνες αυτές είναι λεπτές και έχουν χαμηλή κατανάλωση σε σχέση με τις οθόνες τεχνολογίας καθοδικού σωλήνα (CRT : Cathodic Ray Tube) που χρησιμοποιούνταν παλαιότερα. Άλλοι τύποι οθόνης είναι οι οθόνες που χρησιμοποιούν φωτοδιόδους (LED: Light Emitting Diode) και οι οθόνες πλάσματος.

γ) <u>Ο εκτυπωτής (printer)</u>

Τα πιο γνωστά είδη εκτυπωτών που χρησιμοποιούνται σήμερα είναι: ο εκτυπωτής εκτόξευσης μελάνης (ink jet), ο οποίος εκτοξεύει το μελάνι στο χαρτί και ο εκτυπωτής laser που εξασφαλίζει την καλλίτερη ποιότητα εκτύπωσης. Υπάρχουν μονόχρωμοι και έγχρωμοι εκτυπωτές. Κυριώτερα χαρακτηριστικά των εκτυπωτών είναι η ταχύτητα εκτύπωσης και η ευκρίνεια. Τελευταία έχουν εμφανιστεί στην αγορά και εκτυπωτές τριών διαστάσεων (3D), που “εκτυπώνουν” τρισδιάστατα μοντέλα.

δ) <u>Συσκευές σάρωσης (scanners)</u>

Είναι συσκευές που μπορούν να διαβάσουν χαρακτήρες ή και γραφικές παραστάσεις και να τις καταχωρήσουν στη μνήμη του Η/Υ ως δεδομένα. Υπάρχουν συσκευές ειδικές για σάρωση κειμένου και για σάρωση γραφημάτων (σχεδίων κλπ). Με τη σάρωση χαρακτήρων είναι δυνατόν να επιτευχθεί ανάγνωση χαρακτήρων με μια τεχνική που είναι γνωστή ως οπτική αναγνώριση χαρακτήρων (OCR : Optical Character Recognition).

Στην κατηγορία αυτή ανήκουν και οι αναγνώστες γραμμωτού κώδικα (bar code readers) και κώδικα γρήγορης απόκρισης (QR code : Quick Response code) που διαβάζουν δεδομένα τα οποία παριστάνονται με τη μέθοδο του γραμμωτού κώδικα (δηλαδή παράλληλα τυπωμένες

γραμμές διαφορετικού πάχους) ή δισδιάστατου γραμμωτού κώδικα αντίστοιχα. Οι κώδικες αυτοί χρησιμοποιούνται ευρύτατα σε συσκευασίες καταναλωτικών και άλλων προιόντων,

ε) Σχεδιογράφοι (plotters)

Χρησιμοποιούνται για την παραγωγή σχεδίων και γενικά γραφικών παραστάσεων και απαικονίσεων, μικρού και μεγάλου μεγέθους, Τα δεδομένα βάσει των οποίων γίνεται η σχεδίαση είναι καταχωρημένα στη μνήμη του Η/Υ.

στ) Το ποντίκι (mouse).

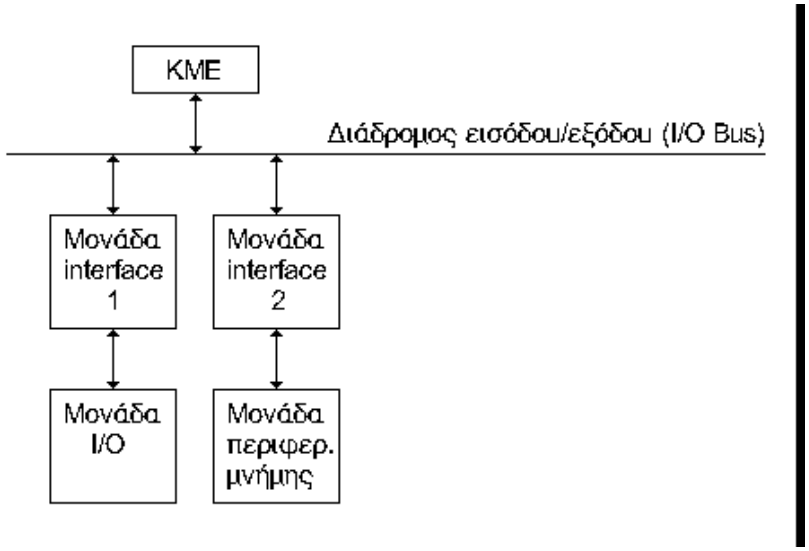
Με το ποντίκι, ο χρήστης μπορεί να “περιηγείται¨ στην οθόνη του υπολογιστή, με τη βοήθεια ενός “δρομέα” (cursor), που αποτυπώνει την κίνηση του ποντικιού στην οθόνη.

Άλλες συσκευές Ε/Ε είναι η ηλεκτρονική γραφίδα (light pen), το μικρόφωνο, η κάμερα, τα ηχεία, τα ακουστικά κ.α.

#### 2.4.2.2 Σύνδεση μονάδων εισόδου-εξόδου με την ΚΜΕ

Όπως είδαμε οι μονάδες εισόδου-εξόδου (Ε/Ε) είναι απαραίτητες για την εισαγωγή και εξαγωγή πληροφοριών προς και από την ΚΜΕ.

Η μονάδα interface I/O χρησιμεύει σαν ενδιάμεσος κρίκος για την μεταφορά πληροφοριών από την Κεντρική Μνήμη ή τους Καταχωρητές της ΚΜΕ προς τις μονάδες Ε/Ε. Ορισμένα μεγάλα υπολογιστικά συστήματα χρησιμοποιούν ειδικούς επεξεργαστές (I/O processors) για την επικοινωνία της ΚΜΕ με τις μονάδες Ε/Ε. Η σύνδεση της ΚΜΕ με τις μονάδες Ε/Ε (αλλά και με τις περιφερειακές μονάδες μνήμης) φαίνεται παραστατικά στο παρακάτω σχήμα :




### 2.4.3 Άλλες περιφερειακές Μονάδες

Με την εξέλιξη των Η/Υ έχουν αναπτυχθεί και διάφορες άλλες περιφερειακές μονάδες, όπως οι μονάδες σύγχρονης επικοινωνίας. Οι μονάδες αυτές χρησιμοποιούνται για να συνδεθούν υπολογιστές μεταξύ τους ή γενικότερα για μετάδοση δεδομένων σε μεγάλες αποστάσεις μέσω τηλεπικοινωνιακών γραμμών. Πιο γνωστή από τις συσκευές αυτές είναι το modem

(διαμορφωτής – αποδιαμορφωτής) που διαμορφώνει το σήμα δηλαδή το μετασχηματίζει με τρόπο ώστε να μπορεί να μεταδοθεί μέσα από τηλεπικοινωνιακή γραμμή σε μεγάλη απόσταση. Η αντίστοιχη συσκευή που υπάρχει στην άλλη άκρη της γραμμής αποδιαμορφώνει το σήμα δηλαδή το επαναφέρει στην αρχική μορφή.

Για τη σύνδεση στο Διαδίκτυο ενός Η/Υ ή ενός απλού δικτύου (όπως μία οικιακή σύνδεση), χρησιμοποιείται μία συσκευή που συνδυάζει modem και δρομολογητή (router). Ο δρομολογητής μεταφέρει πακέτα δεδομένων μεταξύ του Διαδικτύου και του τοπικού (π.χ. οικιακού) δικτύου, περιλαμβάνει δε και διάταξη μεταγωγής (switch). Η διάταξη αυτή παρακολουθεί τα πακέτα δεδομένων που μεταφέρονται και τα κατευθύνει στον προορισμό τους (κόμβος δικτύου, Η/Υ κλπ).

Η σύνδεση μονάδων Ε/Ε και περιφερειακών μονάδων με τον Η/Υ γίνεται στην πράξη ασύρματα αλλά κυρίως ενσύρματα μέσω κατάλληλων καλωδίων και θυρών (ports). Ευρέως χρησιμοποιούμενες σήμερα θύρες είναι η θύρες USB (Universal Serial Bus) και Type C, κυρίως στους προσωπικούς υπολογιστές και τα smartphones, καθώς και η θύρα HDMI (High Definition Multimedia Interface), που χρησιμοποιείται κυρίως για τη σύνδεση οθόνης με τον Η/Υ.

## 2.5 Υπολογιστικά συστήματα, Δίκτυα και τρόποι επεξεργασίας.

### 2.5.1 Υπολογιστικά συστήματα και Δίκτυα

Υπάρχουν σήμερα πολλών ειδών υπολογιστικά συστήματα. Η τελική δομή ενός τέτοιου συστήματος διαμορφώνεται ανάλογα με τις ιδιαίτερες ανάγκες του χρήστη είτε είναι ιδιώτης, επιχείρηση κ.λ.π. Τα πιο σημαντικά παραδείγματα υπολογιστικών Συστημάτων και Δικτύων διασυνδεδεμένων υπολογιστικών συστημάτων είναι τα παρακάτω :

α) Προσωπικός υπολογιστής (PC) Προορίζεται βασικά για έναν χρήστη και είναι απλό υπολογιστικό σύστημα με έναν απλό επεξεργαστή με μικρό κόστος. Περιλαμβάνει οθόνη, πληκτρολόγιο και τις περισσότερες φορές και εκτυπωτή

β) Σύστημα πολλών χρηστών (multi user συστήματα). Αυτά είναι μεγάλα και δαπανηρά συστήματα που περιλαμβάνουν μία μεγάλη και ισχυρή κεντρική μονάδα που εξυπηρετεί ταυτόχρονα πολλές θέσεις εργασίας (τερματικά). Συστήματα της κατηγορίας αυτής είναι οι υπολογιστές μεγάλης κλίμακας (mainframes) και οι υπερυπολογιστές (supercomputers), που μπορεί να περιλαμβάνουν ανεξάρτητους Η/Υ με πολλαπλούς επεξεργαστές ή πολλαπλούς Η/Υ που συνεργάζονται μεταξύ τους.

γ) Τοπικό δίκτυο υπολογιστών (LAN Local Area Network).Ένα τέτοιο σύστημα αποτελείται από πολλούς μικρούς ΗΥ (συνήθως PC) συνδεδεμένους με καλώδια (πιο σπάνια ασύρματα) ώστε να μπορούν να επικοινωνούν μεταξύ τους. Κάθε θέση εργασίας λειτουργεί αυτόνομα αλλά έχει και τη δυνατότητα να επικοινωνεί με όλες τις άλλες για μεταφορά δεδομένων ή για χρήση από κοινού ενός ακριβού περιφερειακού π.χ. ενός εκτυπωτή laser.

δ) Μητροπολιτικό δίκτυο υπολογιστών (MAN : Metropolitan Area Network) και Δίκτυα ευρείας κλίμακας (WAN : Wide Area Networks). Τα δίκτυα αυτά συνδέουν συστήματα Η/Υ που βρίσκονται σε απομακρυσμένα σημεία, π.χ. στα όρια μίας μεγάλης πόλης.

ε) Δίκτυα μεγάλων αποστάσεων (long haul networks) Τα δίκτυα αυτά συνδέουν μεγάλα ή μικρότερα συστήματα Η/Υ διαφόρων τύπων και κατασκευαστών που βρίσκονται σε μεγάλες αποστάσεις μεταξύ τους, ακόμη και σε διαφορετικές χώρες. Παράδειγμα τέτοιου δικτύου είναι το ARPANET, που υπήρξε πρόδρομος του Διαδικτύου.

Η σύνδεση πολλών γεωγραφικά απομακρυσμένων υπολογιστών σε ενιαίο δίκτυο αναφέρεται και ως υπολογιστική πλέγματος (grid computing)

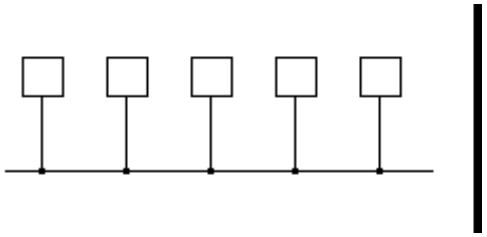
Τέλος, δίκτυο είναι και το Διαδίκτυο (Internet), που είναι και το μεγαλύτερο και πιο εκτεταμένο δίκτυο στον κόσμο.

### 2.5.2 Τοπικά Δίκτυα Υπολογιστών (Local Area Networks : LANs) και τοπολογίες

Στα τοπικά δίκτυα υπολογιστών, οι υπολογιστές που τα αποτελούν (λέγονται και hosts), είναι σχετικά μικροί και διασυνδέονται συνήθως με καλώδια (χαλκού, ομοαξονικά, οπτικών ινών). Οι αποστάσεις μεταξύ των υπολογιστών είναι μικρές (συνήθως στο ίδιο κτίριο). Παρακάτω θα εξηγήσουμε τις βασικές τοπολογίες (αρχιτεκτονικές) των τοπικών δικτύων, που ωστόσο υλοποιούνται και σε μεγαλύτερης κλίμακας δίκτυα.

α) Τοπολογία διαύλου (bus)

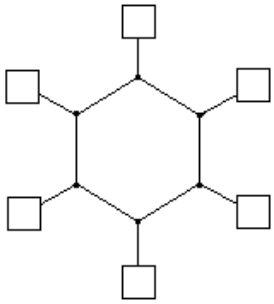
Στο δίκτυο αυτής της μορφής όλοι οι υπολογιστές και οι περιφερειακές μονάδες που αποτελούν το Δίκτυο, συνδέονται πάνω σε ένα κοινό κανάλι (καλωδιακό σύστημα), που λέγεται bus, όπως δείχνει το παρακάτω σχήμα :



Όταν ένας υπολογιστής έχει να στείλει κάποιο μήνυμα πρώτα εξετάζει αν το κοινό bus είναι κατειλημμένο από κάποιον άλλο host, αν είναι περιμένει να τελειώσει, και έπειτα στέλνει το δικό του μήνυμα

β) Δακτύλιος (Ring)

Στο δίκτυο αυτό κάθε υπολογιστής συνδέεται με δύο άλλους, ώστε όλοι μαζί σχηματίζουν ένα δακτύλιο όπως στο σχήμα :

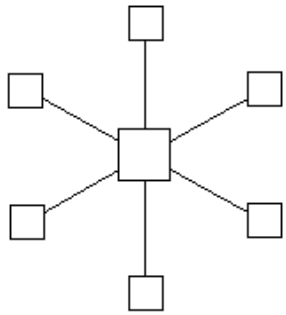


Τα μηνύματα μεταβιβάζονται κατά μήκος του δακτυλίου μέχρις ότου φτάσουν στον προορισμό τους. Συνήθως χρησιμοποιείται ο δακτύλιος με “σκυτάλη” (token ring). Στο δίκτυο αυτού του τύπου, ένα ειδικό μήνυμα που λέγεται “σκυτάλη” (token), κυκλοφορεί γύρω-γύρω στο δακτύλιο. Όποιος υπολογιστής θέλει να στείλει μήνυμα, πρέπει πρώτα να πιάσει τη “σκυτάλη” και μετά να την ξαναβάλει σε κυκλοφορία, αφού εκπέμψει το μήνυμα.

γ) Δίκτυο αστέρας (Star)

Είναι η πιο συνηθισμένη τοπολογία για δίκτυα αρχιτεκτονικής client/server. (πελάτη/διακομιστή), σύμφωνα με την οποία, απαιτείται ένας διακομιστής (server), ο οποίος είναι ένας Η/Υ που μοιράζει τους πόρους του δικτύου στους υπολογιστές των πελατών-χρηστών (clients) του δικτύου. Στο Δίκτυο αυτό εκτός από τον διακομιστή (server), χρησιμοποιείται και μία κεντρική διάταξη μεταγωγής (switch) με την οποία συνδέεται καθένας από τους υπόλοιπους υπολογιστές ή τις περιφερειακές συσκευές του Δικτύου. Η επικοινωνία δύο υπολογιστών θα γίνει μέσω της κεντρικής διάταξης (server & switch), η οποία λειτουργεί με τον μηχανισμό polling, δηλαδή ρωτάει συνεχώς τις συνδεδεμένες μονάδες εάν έχουν να στείλουν μήνυμα και κατόπιν επιτρέπει στην κάθε μία να το στείλει, προκειμένου να αποφευχθεί η σύγκρουση δεδομένων (collision).

Η δομή του είναι όπως στο παρακάτω σχήμα :



Συνηθισμένο είναι να έχουμε σε ένα LAN ορισμένο αριθμό χρηστών υπολογιστών που μπορεί να έχουν κοινή πρόσβαση σε ένα ακριβό περιφερειακό π.χ. έναν εκτυπωτή laser. έναν σαρωτή (scanner) κ.λ.π.
Στην πράξη βέβαια υλοποιούνται και δίκτυα υβριδικής τοπολογίας (hybrid topology).

### 2.5.3 Το Διαδίκτυο (Internet)

Το Internet αποτελεί το μεγαλύτερο και πιο εκτεταμένο Δίκτυο στον κόσμο. Η δημιουργία του ξεκινάει στα τέλη της δεκαετίας του 1950, όταν αναπτύχθηκε στις Η.Π.Α. το Δίκτυο ARPANET για στρατιωτικούς σκοπούς. Για τη ανάπτυξή του στη σημερινή μορφή ήταν αναγκαία, όπως είναι προφανές, η δημιουργία κατάλληλων πρωτοκόλλων επικοινωνίας, Τα πρωτόκολλα αυτά είναι γνωστά με τις ονομασίες TCP/IP και http/www (world wide web, ή απλά web). Σήμερα η χρήση και η λειτουργία των Η/Υ είναι άμεσα συνυφασμένη με τη σύνδεσή τους στο Internet.

### 2.5.4 Τρόποι επεξεργασίας δεδομένων

Υπάρχουν διάφοροι τρόποι επεξεργασίας δεδομένων από ένα υπολογιστικό σύστημα ανάλογα με τις απαιτήσεις και τις ιδιαιτερότητες κάθε εφαρμογής
α) Κατά δέσμες (batch) : Αποτελείται από μία ή περισσότερες εργασίες (jobs) με καθορισμένη σειρά εκτέλεσης. Προηγείται η διαδικασία εισαγωγής των δεδομένων (data entry). Οι εργασίες εκτελούνται όπως και όταν καθορίζει το λειτουργικό σύστημα με κριτήριο τη μεγιστοποίηση της απόδοσης του Η/Υ
β) Απευθείας επικοινωνία (online) : υπάρχει συνεχής και άμεση επικοινωνία χρήστη-υπολογιστή
γ) διαδραστική (intertractive) : Υπάρχει συνεχής επικοινωνία και επιπλέον ο χρήστης ζητά στοιχεία ή δίνει στοιχεία στον υπολογιστή και καθορίζει την περαιτέρω πορεία της επεξεργασίας
δ) πραγματικού χρόνου (real-time) : Η ενημέρωση των αρχείων του συστήματος γίνεται ακριβώς τη στιγμή που γίνονται κάποιες μεταβολές (π.χ. εισαγωγή δεδομένων) στο σύστημα.
ε) Κατανεμημένη (distributed) : Οι χρήστες και οι υπολογιστές είναι γεωγραφικά διασκορπισμένοι για καλύτερη εξυπηρέτηση τοπικά αλλά έχουν τη δυνατότητα μεταξύ τους επικοινωνίας.
στ) Τελευταία αναπτύσσεται ραγδαία το μοντέλο υπολογιστικής νέφους (cloud computing). Στους χρήστες παρέχονται υπηρεσίες αποθήκευσης δεδομένων, επεξεργασίας, χρήσης λογισμικού κ.α. χρησιμοποιώντας μια “δεξαμενή” πόρων μέσω του δικτύου που διασυνδέονται οι χρήστες, και ιδιαίτερα μέσω του Διαδικτύου. Υπηρεσίες όπως η παροχή υποδομής ως υπηρεσία (Infrastructure as a Service - IaaS), λογισμικού ως υπηρεσία (Software as a Service – SaaS), πλατφόρμας ως υπηρεσία (Platform as a Service – PaaS) αποτελούν παραδείγματα αυτής της τεχνικής.

# ΚΕΦΑΛΑΙΟ 3. ΑΡΙΘΜΗΤΙΚΑ ΣΥΣΤΗΜΑΤΑ – ΚΩΔΙΚΕΣ – ΣΤΟΙΧΕΙΑ ΑΛΓΕΒΡΑΣ BOOLE

## 3.1 Αριθμητικά συστήματα

Κάθε αριθμητικό σύστημα χαρακτηρίζεται από έναν αριθμό που λέγεται βάση του συστήματος (π.χ. βάση του δεκαδικού συστήματος είναι το 10). Αν η βάση ενός συστήματος είναι το Ν τότε χρειαζόμαστε Ν διαφορετικά σύμβολα για να παραστήσουμε οποιονδήποτε αριθμό στο σύστημα αυτό. Για να βρούμε ποια ποσότητα εκφράζει ένας αριθμός στον Ν-δικό σύστημα πολλαπλασιάζουμε τα ψηφία του με δυνάμεις του Ν αρχίζοντας από το τελευταίο δεξιά ψηφίο και μηδενική δύναμη, και προσθέτουμε τα επιμέρους αθροίσματα

Ο αριθμός $\alpha_1\alpha_2\alpha_3 \dots \alpha_\kappa$ στον Ν-δικό σύστημα ισούται με

$$(\alpha_1\alpha_2\alpha_3 \dots \alpha_\kappa)_N = \alpha_\kappa * N^0 + \alpha_{\kappa-1} * N^1 + \alpha_{\kappa-2} * N^2 \dots + \alpha_1 * N^{\kappa-1}$$

πχ $267_8 = 7*8^0 + 6*8^1 + 2*8^2 = 7 + 6*8 + 2*64 = 183$ (στο γνωστό μας δεκαδικό σύστημα)

## 3.2 Το δυαδικό σύστημα. Το οκταδικό και το δεκαεξαδικό σύστημα

Στους Η/Υ για την παράσταση των αριθμών (αλλά και άλλων χαρακτήρων όπως θα δούμε στη συνέχεια) χρησιμοποιείται το δυαδικό σύστημα διότι αυτό υλοποιείται εύκολα εκφράζοντας δύο διαφορετικές καταστάσεις (κύκλωμα ανοιχτό ή κλειστό).

Το δυαδικό σύστημα έχει βέβαια ως βάση το 2 και χρησιμοποιεί δύο σύμβολα τα 0 και 1 για την παράσταση ενός αριθμού π.χ. για το δυαδικό αριθμό 1011101 χρησιμοποιώντας τη γενική εξίσωση της προηγούμενης παραγράφου βρίσκουμε οτι αντιστοιχεί στο δεκαδικό αριθμό 93.

Για να μετατρέψουμε ένα δεκαδικό αριθμό σε δυαδικό τον διαιρούμε με 2, στη συνέχεια διαιρούμε το πηλίκο με 2 κ.ο.κ. μέχρι να βρούμε πηλίκο 0. Τα υπόλοιπα όλων των διαδοχικών διαιρέσεων (με αντίστροφη σειρά) μας δίνουν το δυαδικό αριθμό που ζητάμε. ‘Αν πάρουμε π.χ. τον δεκαδικό αριθμό 93

| | | | | | | |
|---|---|---|---|---|---|---|
| 93 | : | 2 | = | 46 | με υπόλοιπο | 1 |
| 46 | : | 2 | = | 23 | » | 0 |
| 23 | : | 2 | = | 11 | » | 1 |
| 11 | : | 2 | = | 5 | » | 1 |
| 5 | : | 2 | = | 2 | » | 1 |
| 2 | : | 2 | = | 1 | » | 0 |
| 1 | : | 2 | = | 0 | » | 1 |

Άρα ο ζητούμενος αριθμός είναι ο 1011101.

Το οκταδικό σύστημα έχει ως βάση το 8 που είναι δύναμη του 2 ($8 = 2^3$).

Η μετατροπή από το οκταδικό στο δυαδικό και αντίστροφα είναι εύκολη αναπτύσσοντας τα οκταδικά ψηφία σε τριάδες δυαδικών ψηφίων ή χωρίζοντας το δυαδικό αριθμό σε τριάδες (άν χρειαστεί προσθέτουμε μηδενικά στην αρχή του) και αντικαθιστώντας κάθε τριάδα με το ισοδύναμο οκταδικο ψηφίο αντίστοιχα

Η ισοδυναμία οκταδικών και δυαδικών ψηφίων είναι η εξής

| οκταδικός | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| δυαδικός | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

παράδειγμα: $3\ 7\ 6_{\ 8} = 011\ 111\ 110_{\ 2}$

ενώ $10101001_2 = 010\ 101\ 001_{\ 2} = 251_8$

Το δεκαεξαδικό σύστημα έχει ως βάση το 16 που είναι και αυτό δύναμη του 2 ($16 = 2^4$). Τα σύμβολα του 16-δικού συστήματος είναι τα 0 1 2 3 4 5 6 7 8 9 A B C D E F - τα γράμματα A B C D E F χρησιμοποιούνται αντί των αριθμών 10 11 12 13 14 15 του δεκαδικού συστήματος.

Για τη μετατροπή δεκαδικών αριθμών σε δυαδικούς και αντίστροφα ισχύει ο κανόνας που αναφέρθηκε για το οκταδικό σύστημα αλλά τώρα έχουμε τετράδες αντί για τριάδες δυαδικών ψηφίων, ενώ η ισοδυναμία δεκαεξαδικών και δυαδικών ψηφίων φαίνεται παρακάτω :

| δεκαεξαδικος | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| δυαδικός | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |

| δεκαεξαδικος | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|
| δυαδικός | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |

παράδειγμα : $96AF_{16} = 1001\ 0110\ 1010\ 1111_{\ 2}$

ενώ $1110010110_2 = 0011\ 1001\ 0110_{\ 2} = 3\ 9\ 6_{\ 16}$

<u>Πράξεις στο δυαδικό σύστημα</u>

Οι τέσσερις πράξεις στο σύστημα γίνονται όπως και στο δεκαδικό

για την πρόσθεση πρέπει να θυμόμαστε τους βασικούς κανόνες

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 και κρατούμενο 1 (για μεταφορά )

ενώ για την αφαίρεση

0 - 0 = 0

1 - 0 = 1

1 - 1 = 0

0 - 1 = 1 με δανεισμό 1 μονάδας

Παραδείγματα

```
  1010        1010        110101
+    1      +   11      + 10110
  1011        1101       1001011

 10101       11011        11010
-   10      -  1111      -  1011
 10011       1100          1111

   101101          110110 | 110
*   1101          -110       1001
   101101            01
  000000           - 0
 101101             011
101101            -  0
1001001001           110
                   -110
                      0
```

### 3.3 Παράσταση δυαδικών αριθμών στον Η/Υ

Για την παράσταση αρνητικών ακέραιων αριθμών στον Η/Υ χρησιμοποιείται η παράσταση συμπληρώματος ως προς ένα (ΠΣ1) και η παράσταση συμπληρώματος ως προς δύο (ΠΣ2).
Στην ΠΣ1 ενός ν-ψηφίου δυαδικού αριθμού αν το πρώτο πιο σημαντικό ψηφίο του είναι 0 τότε ο αριθμός είναι θετικός και το μέτρο του δίνεται από το υπόλοιπα ν-1 ψηφία. Αν το πρώτο ψηφίο του είναι 1 τότε ο αριθμός είναι αρνητικός και το μέτρο του δίνεται από το συμπλήρωμα ως προς 1 των υπολοίπων ν-1 ένα ψηφίων του. Το συμπλήρωμα ως προς 1 ενός δυαδικού αριθμού βρίσκεται αν εναλλάξουμε τα 0 και τα 1, π.χ. το συμπλήρωμα ως προς ένα του 1100101110 είναι 0011010001.
Στην ΠΣ2 ενός ν-ψήφιου δυαδικού αριθμού αν το πρώτο ψηφίο του είναι 0 τότε ο αριθμός είναι θετικός και το μέτρο του δίνεται από τα υπόλοιπα ν - 1 ψηφία. Αν το πρώτο ψηφίο του είναι 1 τότε ο αριθμός είναι αρνητικός και το μέτρο του δίνεται από το συμπλήρωμα ως προς

2 και των ν ψηφίων του. Το συμπλήρωμα ως προς 2 (ή αληθές συμπλήρωμα) ενός δυαδικού αριθμού βρίσκεται αν πάρουμε πρώτα το συμπλήρωμα ως προς 1 του αριθμού και προσθέσουμε το 1.

Για παράδειγμα, το συμπλήρωμα ως προς 2 του 11 00101110 είναι

0011010001 + 1 = 0011010010

Με τους παραπάνω τρόπους παράστασης αρνητικών αριθμών ο Η/Υ διεξάγει πιο εύκολα τις αφαιρέσεις. Για να αφαιρέσουμε τον δυαδικό Β από τον διάδικο Α μπορούμε να προσθέσουμε στον Α το συμπλήρωμα ως προς 2 του Β και να αγνοήσουμε την επιπλέον μονάδα που προκύπτει

Αν για παράδειγμα A=10110101 και B=01010110 και θέλουμε να υπολογίσουμε το A-B :

B = 01010110 άρα συμπλήρωμα του Β ως προς 1 = 10101001

\+ 1

---

συμπλήρωμα του Β ως προς 2 = 10101010

Επομένως 01101011 = A

\+ 10101010 = συμπλήρωμα του Β ως προς 2

---

1)01011111

Άρα το αποτέλεσμα A – B είναι ο αριθμός 1011111

### 3.4 Κωδικοποίηση

Κωδικοποίηση είναι μία διαδικασία με την οποία μετασχηματίζουμε κάποια πληροφορία, ένα σύνολο από σύμβολα από ένα σύστημα σε κάποιο άλλο, σύμφωνα με κάποιον κανόνα. Όλες οι πληροφορίες με τις οποίες τροφοδοτούμε έναν υπολογιστή (αριθμητικά, αλφαβητικά και άλλα σύμβολα) κωδικοποιούνται στο δυαδικό σύστημα. Υπάρχουν διάφοροι τρόποι κωδικοποίησης στο δυαδικό σύστημα, οι λεγόμενοι δυαδικοί κώδικες. Σύμφωνα με αυτούς σε κάθε χαρακτήρα (αλφαβητικό αλφαριθμητικό κ.λ.π.) αντιστοιχούμε ένα δυαδικό αριθμό.

Με έναν κώδικα Ν δυαδικών ψηφίων (bits) μπορούν να παρασταθούν $2^N$ σύμβολα (χαρακτήρες κλπ). Έτσι για την κωδικοποίηση των δεκαδικών ψηφίων (0 ως 9) χρειάζεται ένας κώδικας των 4 bits. Ένας τέτοιος κώδικας είναι ο κώδικας BCD (Binary Coded Decimal). Η αντιστοιχία δεκαδικών δυαδικών ψηφίων είναι οι παρακάτω

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 |

χρησιμοποιώντας των παραπάνω κώδικα μπορούμε να κωδικοποιήσουμε οποιονδήποτε αριθμό του δεκαδικού συστήματος

π.χ. $526_{10} \rightarrow$ 0101 0010 0110

Σε αυτό το σημείο πρέπει να διευκρινίσουμε ότι η κωδικοποίηση ενός αριθμού είναι διαφορετικό πράγμα από τη μετατροπή του στο δυαδικό σύστημα. Έτσι στο προηγούμενο παράδειγμα η μετατροπή του $526_{10}$ δεν θα μας δώσει στο δυαδικό σύστημα τον αριθμό 010100100110.

Με έναν κώδικα των 4 bits όπως ο BCD μπορούμε να παραστήσουμε μέχρι $2^4 = 16$ διαφορετικά σύμβολα και ως εκ τούτου δεν επαρκούν για παράσταση π.χ. των αλφαβητικών χαρακτήρων. Για να κωδικοποιήσουμε περισσότερους χαρακτήρες χρησιμοποιούμε κώδικες με περισσότερα bits. Οι πιο γνωστοί κώδικες είναι οι ακόλουθοι :

UTF-8 (Unicode Transformation Format – 8-bit), που χρησιμοποιεί 8 bits

ASCII (American standard code for information interchange), που χρησιμοποιεί 7 bits + 1 parity bit (ψηφίο ισοτιμίας – θα επεξηγηθεί παρακάτω) .

Άλλοι κώδικες όπως ο EBCDIC (extended binary coded decimal interchange code) που χρησιμοποιεί 8 bits, πρακτικά δεν χρησιμοποιούνται πλέον.

Το δυαδικό ψηφίο ισοτιμίας (parity bit) είναι μία μέθοδος που χρησιμοποιείται για τον έλεγχο σφαλμάτων. Τέτοιες μέθοδοι χρησιμοποιούνται για την ανίχνευση τυχόν σφαλμάτων που μπορεί να προκύψουν κατά την μεταφορά η μετακίνηση μεγάλου όγκου πληροφοριών και τα οποία μπορούν να οδηγήσουν σε απώλεια ή παραποίηση κάποιων χρήσιμων πληροφοριών. Το ψηφίο ισοτιμίας είναι ένα επιπλέον bit που προστίθεται στο τέλος κάποιου κωδικού αυξάνοντας τον κατά 1 bit, χωρίς να αποτελεί μέρος του (πχ ο τετραψήφιος κώδικας BCB γίνεται πενταψήφιος). Υπάρχουν δύο τύποι ισοτιμίας. Σύμφωνα με την άρτια ισοτιμία (even patity bit) σε κάθε κωδικό προσθέτουμε ένα ακόμη δυαδικό ψηφίο έτσι ώστε ο συνολικός αριθμός των άσσων να είναι άρτιος ενώ σύμφωνα με την περιττή ισοτιμία (odd parity bit) προσθέτουμε σε κάθε κωδικό ένα ακόμη δυαδικό ψηφίο ώστε ο αριθμός των άσσων να είναι περιττός.

### 3.5 Στοιχεία άλγεβρας Boole

Η άλγεβρα Boole, που διατυπώθηκε το 1947 από τον Άγγλο μαθηματικό George Boole, χρησιμοποιείται για την περιγραφή και επίλυση λογικών προβλημάτων. Η άλγεβρα Boole χρησιμοποιείται στη σχεδίαση ηλετρονικών κυκλωμάτων του υπολογιστή των οποίων η λειτουργία είναι επίσης σύμφωνη με τους κανόνες της.

Στην άλγεβρα Boole έχουμε προτάσεις οι οποίες παίρνουν δύο δυνατές τιμές και συγκεκριμένα είναι αληθείς ή ψευδείς. Πρόταση μπορέι να είναι μία διατύπωση όπως π.χ.

16 + 2 = 18

23 + 5 = 35

Σήμερα είναι Κυριακή

Ο 3 διαιρεί των 25

κ.λ.π.

Χρησιμοποιούμε το δυαδικό ψηφίο 1 για την τιμή “αληθής” και το ψηφίο 0 για την τιμή ”ψευδής”. Πίνακας αληθείας ονομάζεται ο πίνακας που παρουσιάζει όλες τις δυνατές τιμές μιας πρότασης (συνήθως σε σχέση με κάποια άλλη ή κάποιες άλλες προτάσεις). Χρησιμοποιούμε το δυαδικό ψηφίο 1 για την τιμή “αληθής” και το δυαδικό ψηφίο 0 για την τιμή “ψευδής”.

Πίνακας αληθείας ονομάζεται ο πίνακας που παρουσιάζει όλες τις δυνατές τιμές μιας πρότασης, συνήθως σε σχέση με κάποια άλλη ή κάποιες άλλες προτάσεις.

Αντίστροφη η αρνητική μιας πρότασης Α είναι η πρόταση η οποία είναι αληθής όταν η Α είναι ψευδής και ψευδής όταν η Α είναι αληθής. Αν π.χ .έχουμε την πρόταση Α : “ο 3 διαιρεί τον 25” (ψευδής) τότε η αντίστροφη της Α που συμβολίζεται με Α’ είναι “ο 3 δεν διαιρεί των 25” . (αληθής). Ο πίνακας αληθείας για αυτήν την περίπτωση είναι :

| A | A’ |
|---|---|
| 1 | 0 |
| 0 | 1 |

Συνδέοντας μεταξύ τους προτάσεις με λογικές πράξεις παίρνουμε νέες προτάσεις. Οι αρχικές προτάσεις είναι οι παράγοντες της λογικής πράξης και η τελική πρόταση το αποτέλεσμα.

Οι λογικές πράξεις που χρησιμοποιούνται στην άλγεβρα (εκτός της αντίστροφης που είδαμε πιο πριν) είναι το λογικό άθροισμα και το λογικό γινόμενο. Το λογικό άθροισμα παριστάνεται με Η ή OR ή με τα σύμβολα + ή V ή U. Το αποτέλεσμα της λογικής πράξης Α + Β είναι αληθές όταν τουλάχιστον μία από τις προτάσεις Α, Β είναι αληθής Το λογικό γινόμενο παριστάνεται με ΚΑΙ ή AND και με τα σύμβολα * ή Λ ή ∩. Το αποτέλεσμα της λογικής πράξης Α * Β είναι αληθές όταν και οι δύο προτάσεις Α, Β είναι αληθείς. Ο πίνακας αληθείας των δύο αυτών λογικών πράξεων φαίνεται παρακάτω :

| A | B | A+B | A*B |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |

Η άλγεβρα Boole,όπως και η γνωστή μας άλγεβρα, έχει αξιώματα και θεωρήματα.

**I) Αξιώματα της άλγεβρας Boole**

α) αντιμετάθεση

A+B = B+A

A*B = B*A

β) προσεταιρισμός (απαλοιφή παρενθέσεων)

A+(B+C) = A+(B+C) = A+B+C

A*(B*C) = (A*B)*C = A*B*C

γ) Επιμερισμός

A*(B+C) = A*B+A*C

A+(B*C) = (A+B)*(A+C)

δ) 'Υπαρξη ουδέτερου στοιχείου. Το 0 είναι το ουδέτερο στοιχείο της πρόσθεσης (λογικό άθροισμα) και το 1 του πολλαπλασιασμού (λογικό γινόμενο).

0 + A = A

1 * A = A

ε) για κάθε πρόταση Α υπάρχει η αντίστροφή της (ή συμπλήρωμα) Α' τέτοια ώστε

A + A' = 1

A * A' = 0

Στην άλγεβρα Boole ισχύει η αρχή του δυισμού. Δηλαδή αν μία λογική παράσταση είναι αληθής τότε θα είναι αληθής και η δυική της η οποία προκύπτει αν αντικαταστήσουμε τα μηδενικά με άσσους και αντίστροφα, καθώς και τα ΚΑΙ με Η και αντίστροφα.

Από τα αξιώματα που αναφέρθηκαν προκύπτουν διάφορα θεωρήματα σημαντικότερα από τα οποία είναι τα παρακάτω

α) Θεωρημα De Morgan

(A + B)' = A' * B' και

(A * B)' = A' + B' (σχέση δυική της προηγούμενης)

β) A + 1 = 1 , A + 0 = A

A * 0 = 0 , A * 1 = A (σχέσεις δυικές των προηγούμενων)

γ) A + A = A , A * A = A

δ) A + A * B = A και

A * (A + B) = A (σχέση δυική της προηγούμενης)

ε) A * ( A' + B) = A * B

A + (A' * B) = A + B (σχέση δυική της προηγούμενης)

# ΚΕΦΑΛΑΙΟ 4. ΟΡΓΑΝΩΣΗ ΚΑΙ ΛΕΙΤΟΥΡΓΙΑ ΤΗΣ ΚΜΕ

## 4.1 Μνήμη υπολογιστή, διεύθυνση, θέσεις μνήμης, λέξη, καταχωρητής

Η μνήμη του υπολογιστή είναι ένα σύνολο θέσεων όπου καταχωρούνται (αποθηκεύονται) δεδομένα και προγράμματα (εντολές) όλα κωδικοποιημένα στο δυαδικό σύστημα. Κάθε θέση μνήμης χαρακτηρίζεται από έναν αριθμό μέσω του οποίου μπορεί να προσπελαστεί, δηλαδή να εντοπιστεί μέσα στην Κεντρική Μνήμη. Ο αριθμός αυτός λέγεται διεύθυνση της εν λόγω θέσης μνήμης. Κάθε στοιχείο (δεδομένο) που καταχωρείται σε μία θέση μνήμης παραμένει εκεί μέχρι να αντικατασταθεί κάποιο άλλο. Τα δεδομένα αυτά ανακαλούνται κατά τη διάρκεια της εκτέλεσης ενός προγράμματος από τις άλλες μονάδες ενός Η/Υ για να χρησιμοποιηθούν κατάλληλα.

Κάθε θέση μνήμης αποτελείται από μία σειρά από δυαδικά ψηφία (bits), τα οποία είναι στοιχεία που μπορούν να τεθούν σε δύο διαφορετικές καταστάσεις, συμβολικά 0 και 1. Μία ομάδα 8 bits αποτελεί ένα Byte, ενώ μία ομάδα bits που μπορεί να καταχωρηθεί σε μία θέση μνήμης (και που ο υπολογιστής χειρίζεται σαν “μονάδα δεδομένων”) αποτελεί μία λέξη (word). Μία λέξη περιλαμβάνει έναν αριθμό Bytes, ανάλογα με τον τύπο και την αρχιτεκτονική του υπολογιστή (συνήθως 2, 4 ή 8 Bytes αναλόγως αν έχουμε επεξεργαστή 16 32 ή 64 bits αντίστοιχα).

Καταχωρητής (register) είναι μία ειδική περιοχή μνήμης μικρή σε μέγεθος όπου καταχωρούνται προσωρινά πληροφορίες (δεδομένα). Οι περιοχές αυτές μνήμης είναι ανεξάρτητες από την ΚΜ και οι διάφορες μονάδες του υπολογιστή έχουν ιδιαίτερα γρήγορη πρόσβαση σε αυτές. Το μέγεθος ενός καταχωρητή είναι συνήθως 1 έως 4 λέξεις. Καταχωρητές υπάρχουν τόσο στην ΜΕ (Μονάδα Ελέγχου) όσο και στην ΑΛΜ (Αριθμητική Λογική Μονάδα) ενός υπολογιστή.

## 4.2 Εντολές και προγράμματα. Γλώσσα μηχανής και συμβολική γλώσσα

Όπως είδαμε για την επίλυση ενός προβλήματος πρέπει να εισάγουμε στον υπολογιστή το κατάλληλο πρόγραμμα με τα αντίστοιχα δεδομένα. Ένα πρόγραμμα αποτελείται από εντολές γραμμένες σε κάποια γλώσσα προγραμματισμού. Ο προγραμματισμός των υπολογιστών γίνεται συνήθως, εκτός ειδικών εφαρμογών, στις λεγόμενες γλώσσες υψηλού επιπέδου, που δημιουργήθηκαν από ειδικούς και είναι ανεξάρτητες από τον τύπο του υπολογιστή που χρησιμοποιείται.

Στην πραγματικότητα ο υπολογιστής καταλαβαίνει μόνο τη δική του γλώσσα που λέγεται γλώσσα μηχανής (machine language). Κάθε εντολή αυτής της γλώσσας αντιστοιχεί σε μία “κατάσταση” στο εσωτερικό του υπολογιστή και είναι κωδικοποιημένη στο δυαδικό σύστημα. Η εκτέλεση μιας σειράς τέτοιων εντολών σημαίνει μία εναλλαγή καταστάσεων στο εσωτερικό

του υπολογιστή από την οποία θα προκύψει το επιθυμητό αποτέλεσμα. Τα προγράμματα που γράφουμε σε κάποια γλώσσα υψηλού επιπέδου (source programs) μεταφράζονται στην γλώσσα μηχανής (object programs) με τη βοήθεια ειδικών προγραμμάτων μεταφραστών (compilers) ώστε να μπορούν να εκτελεστούν από τον υπολογιστή. Η γλώσσα μηχανής είναι διαφορετική για κάθε τύπο υπολογιστή και φυσικά πολύ δύσκολη στη χρήση λόγω της δυαδικής κωδικοποίησης. Οι εντολές της γλώσσας μηχανής αποτελούνται βασικά από κωδικούς εντολών και διευθύνσεις. Αν αντικαταστήσουμε τους δυαδικούς αριθμούς που αντιστοιχούν σε εντολές και διευθύνσεις με συμβολικά ονόματα δημιουργούμε συμβολική γλώσσα (assembly language)

### 4.3 Κατηγορίες εντολών

Τις εντολές της γλώσσας μηχανής (ή εντολές μηχανής) ανάλογα με τη λειτουργία που επιτελούν τις χωρίζουμε στις εξής κατηγορίες :

α) Αριθμητικές εντολές. Είναι οι εντολές με τις οποίες γίνονται οι τέσσερις αριθμητικές πράξεις

β) Λογικές εντολές. Με αυτές εκτελούνται λογικές πράξεις της άλγεβρας Boole καθώς και συγκρίσεις

γ) Εντολές διακλάδωσης. Οι εντολές αυτές μεταφέρουν τον έλεγχο του προγράμματος σε κάποιο άλλο σημείο είτε χωρίς συνθήκη είτε υπό συνθήκη, δηλαδή εφόσον πληρείται κάποια λογική συνθήκη

δ) Εντολές μεταφοράς περιεχομένου. Με τις εντολές αυτές γίνεται μεταφορά δεδομένων μεταξύ καταχωρητών ή μεταξύ καταχωρητών και Κεντρικής Μνήμης κ.λ.π.

ε) Εντολές εισόδου-εξόδου. Με τις εντολές αυτές γίνεται μεταφορά δεδομένων από κάποια Περιφερειακή μονάδα στην Κεντρική Μνήμη ή σε κάποιον καταχωρητή και αντίστροφα

στ) άλλες εντολές. Ανάλογα με το συγκεκριμένο τύπο του υπολογιστή έχουμε και κάποιες άλλες εντολές όπως εντολές που σηματοδοτούν την έναρξη ή το τέλος του προγράμματος,

### 4.4 Η μορφή της εντολής

Μία εντολή της γλώσσας μηχανής αποτελείται βασικά από δύο τμήματα

α) τον κωδικό εντολής

β) τις διευθύνσεις

ο κωδικός εντολής καθορίζει το είδος της λειτουργίας που θα εκτελεστεί, ενώ οι διευθύνσεις παριστάνουν τις διευθύνσεις των θέσεων μνήμης τα περιεχόμενα των οποίων θα συμμετάσχουν στη συγκεκριμένη λειτουργία. Ανάλογα με το πλήθος των διευθύνσεων που υπάρχουν έχουμε τρεις κατηγορίες εντολών.

α) εντολές μίας διεύθυνσης

έχουν τη μορφή

| κωδικός εντολής | διεύθυνση 1 |
|---|---|

Στις εντολές αυτές στην περιγραφόμενη από τον κωδικό εντολής λειτουργία θα συμμετέχει και κάποιος καταχωρητής η ταυτότητα του οποίου εμπεριέχεται στον κωδικό εντολής
β) εντολές δύο διευθύνσεων έχουν τη μορφή

| κωδικός εντολής | διεύθυνση 1 | διεύθυνση 2 |
|---|---|---|

Στην περίπτωση αυτή ο κωδικός εντολής καθορίζει π.χ. μία πράξη που θα γίνει μεταξύ των περιεχομένων των δύο διευθύνσεων και το αποτέλεσμα θα αποθηκευτεί σε κάποιον καταχωρητή η ταυτότητα του οποίου εμπεριέχεται στον κωδικό εντολής

γ) εντολές τριών διευθύνσεων έχουν τη μορφή

| κωδικός εντολής | διεύθυνση 1 | διεύθυνση 2 | διεύθυνση 3 |
|---|---|---|---|

Στην περίπτωση αυτή η πράξη που καθορίζει ο κωδικός της εντολής θα γίνει μεταξύ των περιεχομένων των δύο διευθύνσεων και το αποτέλεσμα θα καταχωρηθεί στη θέση μνήμης που καθορίζει η τρίτη διεύθυνση

### 4.5 Μέθοδοι προσδιορισμού διεύθυνσης μνήμης

Για τον προσδιορισμό μιας διεύθυνσης θέσεις μνήμης υπάρχουν διαφορετικοί τρόποι η πιο Συνηθισμένη είναι οι εξής :

α) Άμεσος προσδιορισμός (Direct)
   στην περίπτωση αυτή ο δυαδικός αριθμός που βρίσκεται στο τμήμα της διεύθυνσης της εντολής εκφράζει ακριβώς τη διεύθυνση της θέσης μνήμης το περιεχόμενο της οποίας θέλουμε να προσπελάσουμε.

β) Έμμεσος προσδιορισμός (indirect)
   Στην περίπτωση αυτή ο δυαδικός αριθμός που βρίσκεται στο τμήμα της διεύθυνσης της εντολής εκφράζει τη διεύθυνση της θέσης μνήμης όπου βρίσκεται η διεύθυνση της θέσης μνήμης το περιεχόμενο της οποίας θέλουμε να προσπελάσουμε.

Συνήθως το πρώτο bit του τμήματος της διεύθυνσης της εντολής προσδιορίζει αν έχουμε άμεσο (0) ή έμμεσο (1) προσδιορισμό.

Για παράδειγμα έστω ότι έχουμε υπολογιστή με λέξη 2 bytes = 16 bits Μία εντολή της γλώσσας μηχανής θα έχει 16 bits και στο ότι τα 4 πρώτα αποτελούν τον κωδικό εντολής και

τα υπόλοιπα 12 τη διεύθυνση (εντολές μίας διεύθυνσης). Έστω τυχαία εντολή με κωδικό 0011

η εντολή 0011 0 00000100111




αναφέρεται στη θέση μνήμης με διεύθυνση 39.
Αν έχουμε την εντολή

0011 1 00000100111




και το περιεχόμενο της θέσης μνήμης με διεύθυνση 39 είναι : 0000000001000101

τότε η παραπάνω εντολή αναφέρεται στη θέση μνήμης με διεύθυνση 69.

Ο άμεσος προσδιορισμός είναι ταχύτερος ενώ ο έμμεσος πλεονεκτεί στο ότι με αυτόν μπορούμε να προσπελάσουμε μεγαλύτερη μνήμη (δηλαδή περισσότερες θέσεις μνήμης). Στο προηγούμενο παράδειγμα με τον άμεσο προσδιορισμό απομένουν από τα 16 μόνο 11 bits για τον προσδιορισμό της διεύθυνσης με τα οποία μπορούμε να προσπεράσουμε μέχρι $2^{11}$= 2048 θέσεις μνήμης. Με τον έμμεσο προσδιορισμό διατίθεται μια ολόκληρη θέση μνήμης (16 bits) για τον προσδιορισμό της διεύθυνσης και έτσι μπορούμε να προσδιορίσουμε μέχρι $2^{16}$ = 65536 θέσεις μνήμης.
Υπάρχουν και άλλοι τρόποι προσδιορισμού διεύθυνσης μνήμης σπουδαιότεροι από τους οποίους είναι ο δεικτοδοτούμενος (indexed) και ο σχετικός (relative).

### 4.6 Τρόπος λειτουργίας της ΚΜΕ και καταχωρητές

΄Οπως είδαμε η ΚΜΕ αποτελείται από την ΜΕ (Μονάδα Ελέγχου) και την ΑΛΜ (Αριθμητική Λογική Μονάδα). Εκτός αυτών περιλαμβάνει και έναν αριθμό καταχωρητών. Η βασική λειτουργία της ΚΜΕ είναι να εκτελεί μία ακολουθία εντολών γλώσσας μηχανής που βρίσκονται στην κύρια μνήμη του υπολογιστή (συνήθως σε διαδοχικές θέσεις μνήμης). Για την εκτέλεση κάθε εντολής η ΚΜΕ συμπληρώνει δύο φάσεις που είναι η φάση ανάκλησης και η φάση εκτέλεσης. Και οι δύο μαζί αποτελούν τον κύκλο εντολής. Κατά τη φάση ανάκλησης η εντολή ανακαλείται από την Κεντρική Μνήμη στην Μονάδα Ελέγχου για να αναγνωριστεί και να καθοριστούν οι λεπτομέρειες της περαιτέρω εκτέλεσής της. Κατά τη φάση εκτέλεσης εκτελούνται όλες οι μικρολειτουργίες στις οποίες αναλύεται μία εντολή της γλώσσας μηχανής. Το χρονικό διάστημα που απαιτείται για την εκτέλεση μιας στοιχειώδους μικρολειτουργίας

(για παράδειγμα τη μεταφορά του περιεχομένου ενός καταχωρητή σε άλλον καταχωρητή) ονομάζεται χρόνος κύκλου ΚΜΕ (CPU cycle time).
Ταυτόχρονα η ΚΜΕ εποπτεύει και τη λειτουργία των άλλων μονάδων του υπολογιστή όπως π.χ. τις μονάδες εισόδου/ εξόδου. Αυτό επιτυγχάνεται μέσω του συστήματος διακοπών (interrupts).
Το πλήθος και το είδος των καταχωρητών που χρησιμοποιούνται διαφέρει ανάλογα με τον τύπο του υπολογιστή και οι ονομασίες τους είναι ανάλογες με το είδος της λειτουργίας που εξυπηρετούν. Οι κυριότεροι είναι :
α) ο καταχωρητής δεδομένων μνήμης
β) ο καταχωρητής διευθύνσεων μνήμης
γ) ο καταχωρητής εντολής
δ) ο καταχωρητής διεύθυνσης εντολής (ή μετρητής εντολών)
ε) οι συσσωρευτές

Όταν ένα δεδομένο (μία λέξη) πρέπει να καταχωρηθεί στην Κεντρική Μνήμη ή να διαβαστεί από αυτήν πρέπει πρώτα να μεταφερθεί στον καταχωρητή δεδομένων μνήμης. Η διεύθυνση της θέσης μνήμης στην οποία γίνεται η εγγραφή ή από την οποία γίνεται ανάγνωση καταχωρείται στον καταχωρητή διευθύνσεων μνήμης. Στον καταχωρητή εντολής τοποθετείται κάθε εντολή του προγράμματος και στη συνέχεια αποκωδικοποιείται και εκτελείται από την ΜΕ και την ΑΛΜ. Το περιεχόμενο του καταχωρητή διευθύνσεως εντολής δίνει κάθε φορά τη διεύθυνση της ΚΜ όπου βρίσκεται η επόμενη προς εκτέλεση εντολή.
Τέλος οι συσσωρευτές είναι καταχωρητές που βρίσκονται στην ΑΛΜ στους οποίους καταχωρούνται οι παράγοντες των αριθμητικών και λογικών πράξεων (προερχόμενα από την ΚΜ), καθώς και τα αποτελέσματα των πράξεων αυτών πριν μεταφερθούν στην ΚΜ. Ο αριθμός των συσσωρευτών εξαρτάται από τον τύπο του υπολογιστή.
Εκτός από τους προαναφερθέντες χρησιμοποιούνται και άλλοι καταχωρητές όπως π.χ. καταχωρητές αθροίσματος, γινομένου, πηλίκου, καταχωρητές συγκρίσεως κ.λ.π.

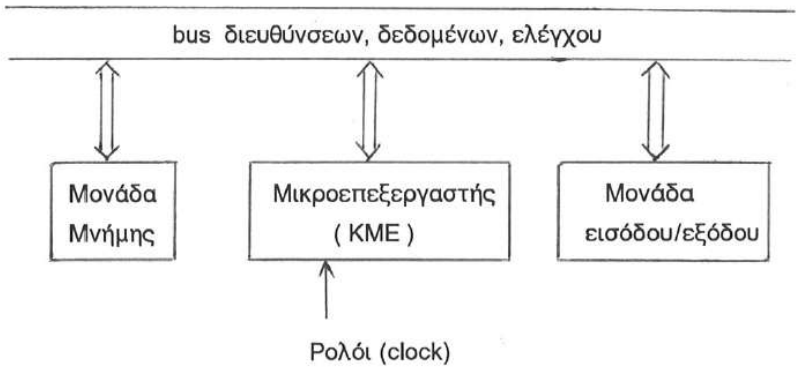
### 4.7 Συμβολική γλώσσα (assembly language)

Όπως είπαμε ήδη μία εντολή της γλώσσας μηχανής είναι μία δυαδική λέξη, δηλαδή μία σειρά δυαδικών ψηφίων για αυτό και είναι δύσκολη στη χρήση. Χρησιμοποιώντας κατάλληλα συμβολικά ονόματα αντί για κωδικούς εντολών και διευθύνσεις μνήμης διευκολυνόμαστε στον προγραμματισμό. Η γλώσσα που δημιουργείται με εντολές αυτής της μορφής λέγεται συμβολική γλώσσα (assembly). Συμβολομεταφραστής (assembler) ονομάζεται ένα ειδικό πρόγραμμα που μετατρέπει τις εντολές της συμβολικής γλώσσας σε εντολές της γλώσσας μηχανής (δυαδικός κώδικας).

### 4.8 Μικροεπεξεργαστές και αρχιτεκτονικές σύγχρονων μικροεπεξεργαστών

Από την εμφάνισή τους, υπάρχει συνεχής εξέλιξη στην τεχνολογία των μικροεπεξεργαστών, με αντίστοιχη αύξηση της πολυπλοκότητας της αρχιτεκτονικής τους (δηλαδή της δομής και οργάνωσής τους). Αυτό σημαίνει οτι αυξάνεται το μήκος της λέξης, η συχνότητα του ρολογιού, ο αριθμός και το μέγεθος των γενικών καταχωρητών που είναι προσπελάσιμοι από τον χρήστη κλπ ενώ υποστηρίζουν και δυνατότητες πολυπρογραμματισμού. Παράλληλα αυξάνει βέβαια και ο χώρος της Κεντρικής Μνήμης ενώ παρέχονται και δυνατότητες για περισσότερους τρόπους προσπέλασης στην Κεντρική Μνήμη.

Ο μικροεπεξεργαστής ενός μικρουπολογιστικού συστήματος επικοινωνεί και ανταλλάσσει πληροφορίες με τις υπόλοιπες μονάδες (μονάδες μνήμης, μονάδες εισόδου/εξόδου), στέλνοντας και λαμβάνοντας διευθύνσεις ή και δεδομένα καθώς και σήματα ελέγχου. Έτσι οι μονάδες του μικρουπολογιστικού συστήματος συνδέονται μεταξύ τους μέσω ενός διαδρόμου (bus) που χωρίζεται στο bus διευθύνσεων (address bus), το bus δεδομένων (data bus) και το bus ελέγχου (control bus). Η γενική διάταξη ενός μικρουπολογιστικού συστήματος φαίνεται στο παρακάτω σχήμα:




Σημαντική εξέλιξη στην αρχιτεκτονική των μικροεπεξεργατών υπήρξαν οι μικροεπεξεργαστές βασισμένοι στην τεχνολογία RISC (Reduced Instruction Set Computers), δηλαδή επεξεργαστές περιορισμένου ρεπερτορίου εντολών, σε αντιδιαστολή με τους συμβατικους επεξεργαστές τεχνολογίας CISC (Complex Instruction Set Computers), δηλαδή επεξεργαστές σύνθετου ρεπερτορίου εντολών. Στους επεξεργαστές RISC έχουμε ένα περιορισμένο σύνολο πρωτογενών εντολών, σε αντίθεση με τους υπολογιστές CISC όπου έχομε μεγάλο πλήθος σύνθετων εντολών με όλους τους συνδυασμούς προσδιορισμού διεύθυνσης μνήμης. Οι πρωτογενείς αυτές εντολές εκτελούνται σε έναν κύκλο ρολογιού της ΚΜΕ. Υπολογιστές Desktop και Servers (ή διακομιστές) είναι σήμερα κυρίως αρχιτεκτονικής CISC, αλλά η αρχιτεκτονική RISC γενικά κερδίζει έδαφος.

‘Ηδη από τη δεκαετία του 1970 αναπτύχθηκαν συστήματα πολλαπλών επεξεργαστών όπου χρησιμοποιούνταν τεχνικές πολυεπεξεργασίας, ώστε να επιτευχθεί πολύ υψηλή απόδοση (ταχύτητα). Ο transputer (transistor computer) ήταν μια τέτοια εξέλιξη, όπου σε ένα

ολοκληρωμένο κύκλωμα (chip) συνδυάζεται η ύπαρξη επεξεργαστή, μνήμης και κυκλωμάτων για επικοινωνία με άλλον transputer χωρίς επιπλέον κυκλώματα διασύνδεσης. Έτσι δημιουργούνται κυκλώματα πολλαπλών transputers οποιουδήποτε μεγέθους. Η γενική αρχιτεκτονική ενός transputer φαίνεται στο παρακάτω σχήμα :




Ωστόσο ήδη από τη δεκαετία του 1990 η τεχνολογία των transputers άρχισε να ξεπερνιέται, καθώς άρχισαν να αναπτύσσονται νεώτερες τεχνολογίες και αρχιτεκτονικές επεξεργαστών, όπως :

- επεξεργαστές πολλαπλών πυρήνων (multi-core processors). Οι επεξεργαστές αυτοί περιλαμβάνουν πολλαπλούς πυρήνες πάνω σε ένα chip, που επιτυγχάνουν παράλληλη επεξεργασία
- κατανεμημένα συστήματα επεξεργασίας (distributed systems.)
- υπολογιστικά συστήματα παράλληλης επεξεργασίας (parallel computing), όπως μονάδες επεξεργασίας γραφικών (GPU - Graphics Processing Unit), οι οποίες περιλαμβάνουν πολυάριθμους μικρούς πυρήνες, που μπορούν να εκτελούν πολλές λειτουργίες ταυτόχρονα.

# ΚΕΦΑΛΑΙΟ 5. ΟΡΓΑΝΩΣΗ ΤΗΣ ΜΝΗΜΗΣ

## 5.1 Η μνήμη του υπολογιστή. Βοηθητική μνήμη.

Σε έναν Η/Υ η Κεντρική Μνήμη (ΚΜ) θεωρείται ως βασικό διαθέσιμο μέσο (central resource) του συστήματος που πρέπει να διατίθεται δυναμικά στα προγράμματα και στους χρήστες. Για το στο σκοπό αυτό είναι αναγκαίο να υπάρχουν αποτελεσματικοί μηχανισμοί για τη διαχείριση (καθώς και για την προστασία) της Κεντρικής Μνήμης. Αυτό είναι έργο του Λειτουργικού Συστήματος. Στη συνέχεια θα δούμε ορισμένες τεχνικές που χρησιμοποιούνται για την καλλίτερη διαχείριση της μνήμης.

Όλοι οι υπολογιστές διαθέτουν εκτός από την Κεντρική Μνήμη η οποία επικοινωνεί άμεσα με την ΚΜΕ και κάποιες μονάδες βοηθητικής μνήμης. Στην κύρια μνήμη βρίσκονται μόνο τα προγράμματα και τα δεδομένα που χρησιμοποιούνται την συγκεκριμμένη στιγμή από τον επεξεργαστή. Όλες οι υπόλοιπες πληροφορίες αποθηκεύονται στη βοηθητική μνήμη και μεταφέρονται στην κεντρική μνήμη όταν αυτό ζητηθεί.

Οι πιο συνηθισμένες μονάδες βοηθητικής μνήμης που χρησιμοποιούνται είναι οι μαγνητικοί δίσκοι, τα μαγνητικά τύμπανα και οι μαγνητικές ταινίες. Τα πιο σημαντικά χαρακτηριστικά μιας διάταξης μνήμης είναι ο χρόνος προσπέλασης, ο ρυθμός μεταφοράς, η χωρητικότητα και το κόστος. Χρόνος προσπέλασης είναι ο μέσος χρόνος που απαιτείται για να εντοπιστεί κάποια θέση μνήμης και να ληφθεί το περιεχόμενό της. Στις διατάξεις μνήμης που περιλαμβάνουν ηλεκτρομηχανικές διατάξεις ο χρόνος πρσπέλασης διαιρείται στο χρόνο αναζήτησης και το χρόνο μεταφοράς. Χρόνος αναζήτησης είναι ο χρόνος που χρειάζεται για να εντοπίσει ο μηχανισμός ανάγνωσης εγγραφής μια θέση μνήμης, ενώ χρόνος μεταφοράς είναι ο χρόνος που απαιτείται για να μεταφερθούν τα δεδομένα από και πρός τη διάταξη μνήμης. Ο χρόνος αναζήτησης είναι συνήθως αρκετά μεγαλύτερος από το χρόνο μεταφοράς και για το λόγο αυτό η βοηθητική μνήμη οργανώνεται σε εγγραφήματα, τα οποία είναι ένα σύνολο λέξεων. Η εγγραφή και η ανάγνωση πρός και από τη μνήμη γίνεται πάντα με ολόκληρα εγγραφήματα, Ρυθμός μεταφοράς είναι ο αριθμός λέξεων που μπορεί να μεταφερθεί ανα δευτερόλεπτο (αφού εντοπιστεί η αρχή του εγγραφήματος).

Στα μέσα μαγνητικής εγγραφής, όπως οι μαγνητικοί δίσκοι (discs), τα bits εγγράφονται και στη συνέχεια διαβάζονται ως μαγνητικές κουκίδες πάνω σε μια επιφάνεια, καθώς η επιφάνεια αυτή περνάει από την κεφαλή εγγραφής – ανάγνωσης. Στο μαγνητικό δίσκο, όπως είδαμε, τα bits αποθηκεύονται στη μαγνητική επιφάνεια σαν μαγνητικές κουκίδες κατά μήκος ομόκεντρων κύκλων που ομνομάζονται ίχνη ή άτρακτοι (tracks). Τα ίχνη διαιρούνται σε τμήματα που ονομάζονται τομείς (sectors). Ο τομέας περιέχει ορισμένο αριθμό bytes και είναι συνήθως η ελάχιστη ποσότητα πληροφορίας που μεταφέρεται κατά την ανάγνωση/εγγραφή της βοηθητικής μονάδας μνήμης. Ένα σύστημα μαγνητικού δίσκου προσπελαύνεται με bits

διεύθυνσης τα οποία προσδιορίζουν τον αριθμό δίσκου, την επιφάνεια του δίσκου, τον αριθμό ίχνους και τον τομέα ίχνους. Οι κεφαλές εγγραφής/ανάγνωσης μετακινούνται πάνω στο ίχνος που καθορίζει η διεύθυνση και περιμένουν μέχρι ο περιστρεφόμενος δίσκος να φέρει τον καθορισμένο τομέα ακριβώς από κάτω. Η βοηθητική μνήμη έχει μεγάλη χωρητικότητα και χαμηλό κόστος σε σχέση με την Κεντρική Μνήμη αλλά έχει μικρή ταχύτητα προσπέλασης.

### 5.2 Γρήγορη Μνήμη (cache memory)

Κατα την εκτέλεση ενός προγράμματος στον υπολογιστή έχει παρατηρηθεί το φαινόμενο της τοπικότητας της αναφοράς στη μνήμη. Αυτό σημαίνει οτι οι αναφορές στη μνήμη γίνονται κατά το μεγαλύτερο μέρος τους σε λίγες εντοπισμένες περιοχές της μνήμης. Αυτό γίνεται κατανοητό αν σκεφτούμε οτι στα προγράμματα έχουμε βρόγχους και υπορουτίνες που καλούνται συχνά ενώ και όσον αφορά τα δεδομένα έχουμε συχνές αναφορές σε περιοχές μνήμης που που βρίσκονται πίνακες και άλλες ενότητες δεδομένων, Το αποτέλεσμα είναι λοιπόν να γίνονται συχνές αναφορές σε κάποιες εντοπισμένες μικρές περιοχές μνήμης ενώ στο υπόλοιπο μέρος της μνήμης οι αναφορές είναι σπάνιες.

Αν οι περιοχές της μνήμης με προγράμματα και δεδομένα όπου γίνονται συχνές αναφορές τοποθετηθούν σε μια μικρή και γρήγορη μνήμη ο μέσος χρόνος προσπέλασης στη μνήμη μπορεί να μειωθεί και μαζί του ο χρόνος εκτέλεσης του προγράμματος. Η γρήγορη μνήμη (cache memory) είναι μια τέτοια μνήμη. Η προσπέλαση στη γρήγορη μνήμη είναι 5 – 10 φορές γρηγοροτερη απ'ότι στην Κεντρική Μνήμη. 'Οταν η ΚΜΕ πρόκειται να προσπελάσει, τη μνήμη εξετάζει πρώτα τη γρήγορη μνήμη. Άν η λέξη που αναζητείται βρεθεί εκεί διαβάζεται, διαφορετικά εξετάζεται η Κεντρική Μνήμη. Μαζί με την ανάγνωση της λέξης γίνεται μεταφορά μιάς ενότητας λέξεων από την Κεντρική στη γρήγορη μνήμη (η ενότητα αυτή θα περιέχει βέβαια τη λέξη που μόλις αναζητήθηκε). Έτσι όταν ξαναγίνει αναφορα στην ίδια λέξη ή σε λέξη που ανήκει στην ενότητα που μόλις μεταφέρθηκε. η αναφορά στη ίδια λέξη ή σε λέξη που ανήκει στην ενότητα που μεταφέρθηκε, η προσπέλασή της θα είναι πλέον πολύ γρήγορη μέσα στη γρήγορη μνήμη.

Η απόδοση της γρήγορης μνήμης μετριέται με ένα μέγεθος που ονομάζεται λόγος επιτυχίας. Επιτυχία (ή ευστοχία) λέμε οτι έχουμε όταν μια λέξη που αναζητείται από την ΚΜΕ βρεθεί στη γρήγορη μνήμη, διαφορετικά λέμμε οτι έχουμε αποτυχία (ή αστοχία). Στη δεύτερη περίπτωση η λέξη θα βρεθεί στην Κεντρική Μνήμη. Ο λόγος επιτυχίας ισούται με το λόγο επιτυχιών προς το σύνολο των αναφορών της ΚΜΕ στη μνήμη (επιτυχίες και αποτυχίες). Ο λόγος επιτυχίας έχει μετρηθεί πειραματικά και έχουν καταγραφεί τιμές 0,9 ή και μεγαλύτερες.. Όταν ο λόγος επιτυχίας είναι αρκετά μεγάλος (δηλαδή κοντά στο 1) σημαίνει οτι τις περισσότερες φορές η

ΚΜΕ βρίκει τις λέξεις που αναζητά στην γρήγορη μνήμη και δεν χρειάζεται να τις αναζητήσει στην Κεντρική Μνήμη.

Ο μετασχηματισμός των δεδομένων από την Κεντρική Μνήμη στη γρήγορη μνήμη ονομάζεται διαδικασία απεικόνισης. Υπάρχουν διάφοροι τύποι διαδικασιών απεικόνισης. Οι πιο συνηθισμένοι είναι :

α) Η συσχετιστική απεικόνιση

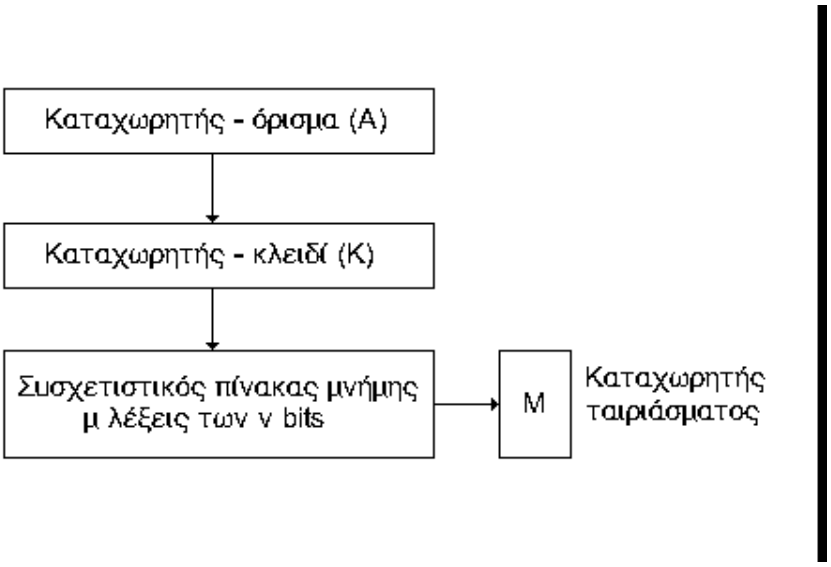
β) Η άμεση απεικόνιση

γ) Η απεικόνιση συσχέτισης συνόλου

### 5.3 Η συσχετιστική μνήμη (associative memory)

Η συσχετιστική μνήμη είναι ένας τύπος μνήμης με την οποία επιτυγχάνουμε την αναζήτηση και ανάκτηση δεδομένων με βάση κάποιο περιεχόμενο, ή μέρος περιεχομένου των δεδομένων, αντί της αναζήτησης στη συμβατική μνήμη που γίνεται με βάση τη διεύθυνση των δεδομένων.

Οι μονάδες που αποτελούν μια συσχετιστική μνήμη φαίνονται στο παρακάτω σχήμα :




Ο συσχετιστικός πίνακας μνήμης αποτελείται από μ λέξεις κάθε μια από τις οποίες έχει ν bits. Ο καταχωρητής – όρισμα (A) και ο καταχωρητής κλειδί (K) έχουν ν bits (που αντιστοιχούν στα ν bits κάθε λέξης), και ο καταχωρητής ταιριάσματος (M) έχει μ bits ένα για κάθε λέξη του συσχετιστικού πίνακα μνήμης. Κάθε λέξη της μνήμης συγκρίνεται με το περιεχόμενο του καταχωρητή – ορισμα. Όταν μια λέξη ταιριάζει με το περιεχόμενο του Α. (δηλαδή όλα τα αντίστοιχα bits ταιριάζουν) το αντίστοιχο bit του καταχωρητή ταιριάσματος γίνεται 1. Στη συνέχεια διαβάζονται όλες εκείνες οι λέξεις για τις οποίες όλα τα αντίστοιχα bits του καταχωρητή Μ είναι 1.

Ο καταχωρητής κλειδί (Κ) παρέχει μια μάσκα για την επιλογή ενός πεδίου μέσα στον Α. Δηλαδή από το σύνολο των bits του Α επιλέγονται μόνο εκείνα που στις αντίστοιχες θέσεις του Κ υπάρχει 1 και συγκρίνονται με τα αντίστοιχα bits των λέξεων μνήμης. Για παράδειγμα , έστω τα περιεχόμενα των Α και Κ είναι :

Α : 1011 1110 1111

Κ : 1111 0000 0000

Μόνο τα τέσσερα πιο αριστερά bits του Α θα συγκριθούν με τις λέξεις μνήμης διότι τα αντίστοιχα bits του Κ είναι μονάδες.

Έτσι από τις παρακάτω θέσεις μνήμης

Λέξη 1 : 1011 1001 1011

Λέξη 2 : 1010 1110 0001

Η πρώτη ταιριάζει ενώ η δεύτερη δεν ταιριάζει.

### 5.4 Ιδεατή μνήμη (virtual memory)

Σε έναν υπολογιστή τα προγράμματα και τα δεδομένα αποθηκεύονται αρχικά στη βοηθητική μνήμη και στη συνέχεια μεταφέρονται κατά τμήματα στην ΚΜ με τη σειρά που ζητούνται από την ΚΜΕ. Η ιδεατή μνήμη είναι μια τεχνική που επιτρέπει στο χρήστη να προγραμματίζει σαν να είχε στη διάθεσή του χώρο μνήμης ίσο με το σύνολο της βοηθητικής μνήμης. Κάθε διεύθυνση που ζητείται πηγαίνει από την ιδεατή διεύθυνση (virtual address) μέσω μιας απεικόνισης διευθύνσεων σε μια πραγματική διεύθυνση της ΚΜ. Έτσι ένα σύστημα ιδεατής μνήμης περιλαμβάνει κάποιον μηχανισμό μετάφρασης των διευθύνσεων που παράγονται από ένα πρόγραμμα σε πραγματικές διευθύνσεις της ΚΜ. Αυτό γίνεται με τη βοήθεια ενός πίνακα απεικόνισης.

Το σύνολο των ιδεατών διευθύνσεων αποτελούν το χώρο διευθύνσεων. Κάθε διεύθυνση της ΚΜ ομομάζεται φυσικη διεύθυνση. Η ΚΜ λέγεται και χώρος μνήμης.

Απλοποιημένο παράδειγμα: ‘Εστω υπολογιστής με χωρητικότητα κεντρικής μνήμης 32 Κ = $2^{15}$ θέσεις μνήμης ενώ η βοηθητική μνήμη του έχει χωρητικότητα 1024 Κ = $2^{20}$ θέσεις μνήμης. Η βοηθητική μνήμη είναι 32 φορές μεγαλύτερη από την ΚΜ. Για την παράσταση μιάς ιδεατής διεύθυνσης απαιτούνται 20 bits ενώ για την παράσταση μιας φυσικής διεύθυνσης απαιτούνται 15 bits. Το πεδίο διεύθυνσης των εντολών της γλώσσας μηχανής θα έχει 20 bits για την παράσταση του συνόλου των ιδεατών διευθύνσεων.

#### 5.4.1 Απεικόνιση διευθύνσεων

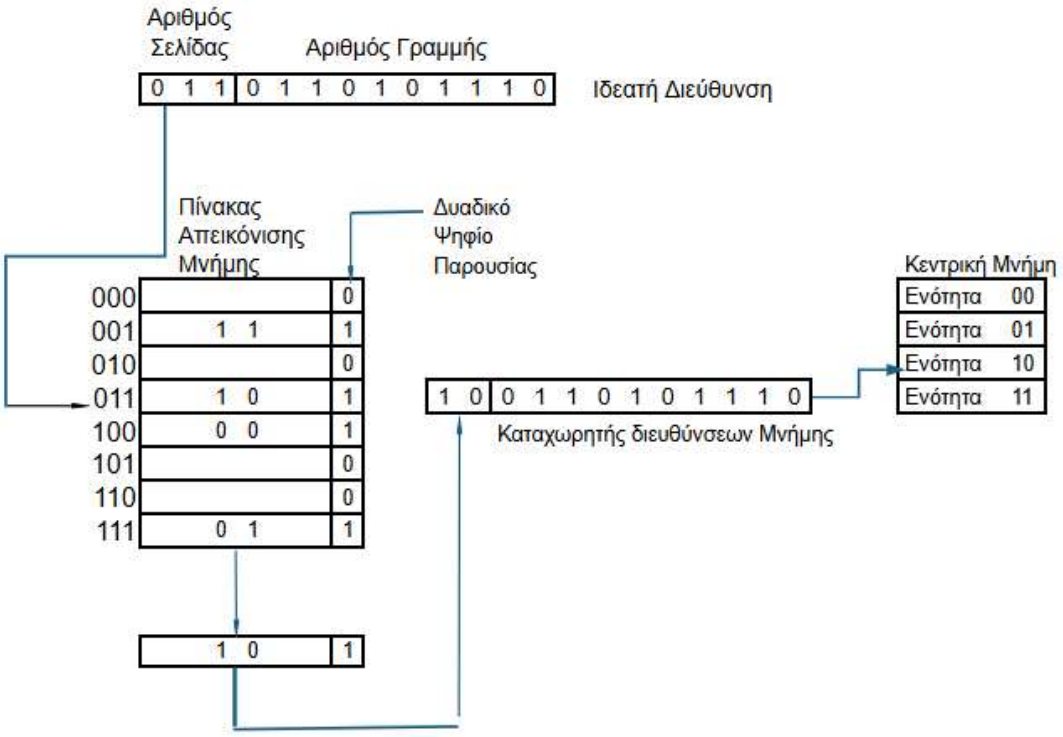
Θεωρούμε οτι ο χώρος της ΚΜ διαιρείται σε κομμάτια ίσου μεγέθους που λέγονται ενότητες (αποτελούνται από έναν αριθμό λέξεων) ενώ ο χώρος διευθύνσεων διαιρείται σε σελίδες που έχουν ίδιο μέγεθος με τις ενότητες. Στο προηγούμενο παράδειγμα που χρησιμοποιήσαμε αν

το μέγεθος μιας σελίδας ή μιας ενότητας είναι 1 K = 1024 θέσεις μνήμης (λέξεις) τότε ο χώρος διευθύνσεων διαιρείται σε 1024 σελίδες ενώ η ΚΜ σε 32 ενότητες.

Για να διευκολυνθεί η απεικόνιση από το χώρο διευθύνσεων στο χώρο μνήμης, θεωρούμε οτι κάθε ιδεατή διεύθυνση περιλαμβάνει τον αριθμό σελίδας και τον αριθμό γραμμής μέσα στη σελίδα. Αν θεωρήσουμε ένα πιο απλό παράδειγμα με χώρο διευθύνσεων 8 K = $2^{13}$ θέσεις μνήμης και χώρο μνήμης 4 K = $2^{12}$ θέσεις μνήμης και χωρίσουμε τον καθένα σε τμήματα των 1 K θέσεων, τότε ο χώρος διευθύνσεων θα αποτελέιται από 8 σελίδες και ο χώρος μνήμης από 4 ενότητες. Κάθε στιγμή μέχρι 4 σελίδες του χώρου διευθύνσεων μπορούν να βρίσκονται στην ΚΜ σε οποιεσδήποτε από τις 4 ενότητες. Η εικονική διεύθυνση σε αυτό το σύστημα θα αποτελείται από 13 bits εκ'των οποίων τα τρια πιο σημαντικά θα προσδιορίζουν τον αριθμό σελίδας (μέχρι οκτώ σελίδες) και τα δέκα λιγότερο σημαντικά τον αριθμό γραμμής (δηλ. τη λέξη) μέσα στη σελίδα. Η διεύθυνση γραμμής είναι η ίδια στο χώρο διευθύνσεων και στο χώρο μνήμης. Επομένως χρειάζεται μια απεικόνιση από τον αριθμό σελίδας στον αριθμό ενότητας.

Η απεικόνιση αυτή γίνεται με τη βοήθεια ενός πίνακα απεικόνισης μνήμης. Ο πίνακας αυτός αποτελειται από οκτώ λέξεις (δηλ. όσες οι σελίδες). Η διεύθυνση κάθε λέξης σε αυτόν τον πίνακα σελίδων υποδηλώνει τον αριθμό σελίδας ενώ το περιεχόμενο της δίνει τον αριθμό ενότητας όπου αυτή η σελίδα αποθηκεύεται στην ΚΜ. Υπάρχει επίσης σε κάθε λέξη και ένα bit παρουσίας που υποδηλώνει αν η αντίστοιχη σελίδα έχει μεταφερθεί στην ΚΜ (1) ή όχι (0). Έτσι όταν διαβάζεται μια εικονική διεύθυνση, τα τρία πρώτα bits προσδιορίζουν τον αριθμό σελίδας και ταυτόχρονα μία διεύθυνση στον πίνακα απεικόνισης. Το περιεχόμενο της αντίστοιχης λέξεις του πίνακα, αν το bit της παρουσίας είναι 1, μαζί με τον αριθμό γραμμής (το δεύτερο τμήμα της εικονικής διεύθυνσης) αποτελούν την αντίστοιχη φυσική διεύθυνση και γι'αυτό μεταφέρονται στον καταχωρητή διευθύνσεων μνήμης. Άν το bit παρουσίας είναι 0, ενεργοποιείται διαδικασία μεταφοράς της ζητούμενης σελίδας από την βοηθητική στην ΚΜ (μέσω του Λειτουργικού Συστήματος) και προσδιορίζεται με τον ίδιο τρόπο η η φυσική διεύθυνση στην ΚΜ. Ο πίνακας απεικόνισης μπορεί να υλοποιηθεί και με συσχετιστική μνήμη για μεγαλύτερη αποδοτικότητα.

Στο παρακάτω σχήμα απεικονίζεται ένα απλοποιημένο παράδειγμα συστήματος ιδεατής μνήμης με χώρο διευθύνσεων 8 σελίδων και χώρο μνήμης 4 ενοτήτων.


Αριθμός Σελίδας
Αριθμός Γραμμής
0 1 1 0 1 1 0 1 0 1 1 1 0
Ιδεατή Διεύθυνση
Πίνακας Απεικόνισης Μνήμης
Δυαδικό Ψηφίο Παρουσίας
000 0
001 1 1 1
010 0
011 1 0 1
100 0 0 1
101 0
110 0
111 0 1 1
1 0 1
1 0 0 1 1 0 1 0 1 1 1 0
Καταχωρητής διευθύνσεων Μνήμης
Κεντρική Μνήμη
Ενότητα 00
Ενότητα 01
Ενότητα 10
Ενότητα 11

# ΚΕΦΑΛΑΙΟ 6. ΤΟ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ

## 6.1 Γενική παρουσίαση του λειτουργικού συστήματος

### 6.1.1 Εισαγωγή

Όπως είδαμε το λειτουργικό σύστημα (Λ.Σ.) ενός υπολογιστικού συστήματος ανήκει στα προγράμματα συστήματος (system software) . Είναι ένα σύνολο προγραμμάτων που ενεργεί ως ενδιάμεσος ανάμεσα στο υλικό (hardware) του υπολογιστή και τα προγράμματα εφαρμογών που δουλεύει ο χρήστης ή οι χρήστες. Επίσης ελέγχει και συντονίζει όλες τις εργασίες που γίνονται στον υπολογιστή Το Λ.Σ. βοηθά το χρήστη με πολλούς τρόπους αλλά η παρουσία του δεν γίνεται άμεσα αντιληπτή από αυτόν. Στην αρχή της εμφάνισης των Η/Υ δεν υπήρχε λειτουργικό σύστημα και ο χρήστης έπρεπε να προγραμματίζει απευθείας στον υπολογιστή στη γλώσσα μηχανής και να εκτελέσει πολλές από τις εργασίες που εκτελεί σήμερα το λειτουργικό σύστημα.

Βασικός σκοπός του λειτουργικού συστήματος είναι αφενός να διευκολύνει τον χρήστη και αφετέρου να εξασφαλίσει την όσο το δυνατόν πιο αποδοτική χρήση του υπολογιστή. Οι σπουδαιότερες από τις εργασίες που εκτελούνται το λειτουργικό σύστημα είναι

- Μεταφέρει τις εντολές και τα αιτήματα του χρήστη προς τον Η/Υ
- Μεταφέρει μηνύματα του Η/Υ προς τον χρήστη π.χ. πληροφορίες για την κατάσταση του συστήματος, σχετικά με προβλήματα λειτουργίας που παρουσιάζονται κλπ
- Ελέγχει και διαχειρίζεται την κεντρική μνήμη (Κ.Μ.)
- Ελέγχει τη λειτουργία της Κ.Μ.Ε. φροντίζοντας παράλληλα για την κατανομή του χρόνου της στις διάφορες στους χρήστες (σε συστήματα πολλαπλών χρηστών) ή στις διάφορες διεργασίες (processes)
- Διαχειρίζεται τις μονάδες εισόδου/εξόδου ελέγχοντας την είσοδο των δεδομένων και την έξοδο των αποτελεσμάτων
- Διαχειρίζεται αρχεία του συστήματος
- Ελέγχει και παρακολουθεί την εκτέλεση των προγραμμάτων των χρηστών
- Φροντίζει για την έναρξη της λειτουργίας του υπολογιστή
- Εξασφαλίζει λειτουργίες ασφαλείας και ελέγχου

Υπάρχουν και άλλες επιμέρους εργασίες που εκτελεί το λειτουργικό σύστημα που είναι δύσκολο να αναφερθούν μία προς μία, δεδομένου ότι το λειτουργικό σύστημα είναι ένα περίπλοκο και μεγάλο σύνολο προγραμμάτων.

### 6.1.2 Πυρήνας και φλοιός του λειτουργικού συστήματος

Από το σύνολο των προγραμμάτων του λειτουργικού συστήματος υπάρχει ένα σχετικά μικρό τμήμα που λέγεται πυρήνας (kernel). Ο πυρήνας παραμένει διαρκώς φορτωμένος στην

κεντρική μνήμη κατά τη διάρκεια της λειτουργίας του υπολογιστή, εκτελεί τις βασικές λειτουργίες και κάνει προσπέλαση σε άλλα τμήματα του λειτουργικού συστήματος. Το υπόλοιπο και μεγαλύτερο τμήμα παραμένει στην βοηθητική μνήμη και διάφορα μέρη του καλούνται από τον πυρήνα στην κεντρική μνήμη όποτε χρειάζεται. Όταν ο Η/Υ τεθεί σε λειτουργία εκτελείται ένα μικρό πρόγραμμα που συνήθως είναι αποθηκευμένο σε μνήμη ROM (και συνήθως αναφέρεται ως monitor). Η πρώτη λειτουργία του προγράμματος αυτού είναι να ελέγξει την μνήμη, δηλαδή να επιβεβαιώσει ότι αυτή μπορεί να χρησιμοποιηθεί. Στη συνέχεια μεταβαίνει στο πρώτο τμήμα του δίσκου συστήματος, όπου είναι αποθηκευμένο το λειτουργικό σύστημα και φορτώνει στην Κ.Μ. ένα μικρό πρόγραμμα, το οποίο αναλαμβάνει να φορτώσει και τον υπόλοιπο πυρήνα στην Κ.Μ. Η διαδικασία αυτή λέγεται εκκίνηση (booting) του συστήματος και όταν τελειώσει, ο χρήστης μπορεί να δώσει εντολές του λειτουργικού συστήματος, να φορτώσει από τη βοηθητική μνήμη, να τρέξει προγράμματα κ.λ.π.

Για να περιγράφει μία εργασία (job) όπως π.χ. μετάφραση η εκτέλεση ενός προγράμματος, χρησιμοποιείται μία ειδική γλώσσα πού διαφέρει από υπολογιστή σε υπολογιστή και λέγεται γλώσσα ελέγχου εργασιών (job control language – JCL). Με τη βοήθεια της γλώσσας αυτής υποβάλλονται οι εργασίες στον Η/Υ και ελέγχεται η εκτέλεσή τους. Σε πολλά λειτουργικά συστήματα οι γλώσσες αυτές έχουν εξελιχθεί στους λεγόμενους φλοιούς (shells). Ο πυρήνας του λειτουργικού συστήματος στην πραγματικότητα δεν γίνεται αντιληπτός από το χρήστη ο οποίος επικοινωνεί με τον Η/Υ μέσω του φλοιού. Γενικά ένα πρόγραμμα που παρακολουθεί τις ενέργειες του χρήστη ερμηνεύει τις εντολές του και τις μεταφέρει για εκτέλεση αναφέρεται ως επεξεργαστής εντολών (command processor).

### 6.1.3 Προγράμματα του λειτουργικού συστήματος

Άλλα προγράμματα του λειτουργικού συστήματος που εκτελούν επιμέρους λειτουργίες είναι :

- Τα προγράμματα ελέγχου εισόδου/εξόδου (input/output control) τα οποία ενεργοποιούν την κατάλληλη κάθε φορά συσκευή εισόδου/εξόδου (ή γενικότερα περιφερειακή μονάδα) και παρακολουθούν τη λειτουργία των συσκευών αυτών. Στους προσωπικούς υπολογιστές το πρόγραμμα αυτό παραμένει στη μνήμη ROM, και παλαιότερα ήταν γνωστό ως BIOS (Basic Input Output System) αλλά σήμερα χρησιμοποιείται ως επί το πλείστον ο διάδοχός του UEFI (Unified Extensible Firmware Interface).
- Τα βοηθητικά προγράμματα διαχείρισης αρχείων χρησιμοποιούνται πιο εύκολα και αποδοτικά και εκτελούν εργασίες όπως διαγραφή, αντιγραφή, συγχώνευση, ταξινόμηση αρχείων κ.λ.π.
- Το πρόγραμμα βιβλιοθήκης διατηρεί έναν κατάλογο από συχνά χρησιμοποιούμενα προγράμματα ή ρουτίνες τα οποία συνήθως χρησιμοποιούνται στον υπολογισμό μαθηματικών και άλλων συναρτήσεων, ελέγχουν περιφερειακές μονάδες κ.λ.π.

Σε ένα λειτουργικό σύστημα υπάρχει ειδικό πρόγραμμα για το χειρισμό των διακοπών (interrupts). Μία από τις κυριότερες πηγές διακοπών είναι οι Μονάδες εισόδου/εξόδου (πληκτρολόγιο, οθόνη κ.λ.π.) Για να εκτελεστεί μία τέτοια εντολή πρέπει ο έλεγχος να περάσει στο λειτουργικό σύστημα. Δημιουργείται λοιπόν μία διακοπή (interrupt) και το λειτουργικό σύστημα κρατάει ορισμένες πληροφορίες που αφορούν το πρόγραμμα του οποίου η λειτουργία διακόπηκε (εντολή που συνέβη η διακοπή, περιεχόμενα καταχωρητών κλπ). Στη συνέχεια εκτελεί την εντολή που προκάλεσε τη διακοπή και τέλος δίνει τον έλεγχο σε κάποιο άλλο πρόγραμμα που βρίσκεται φορτωμένο στην Κεντρική Μνήμη για να συνεχιστεί η εκτέλεσή του.

### 6.1.4 Καταμερισμός χρόνου (time sharing)

Η τεχνική αυτή εφαρμόζεται σε πολλά λειτουργικά συστήματα. Σύμφωνα με αυτή, ο χρόνος διαιρείται σε χρονικές περιόδους Τ και κάθε χρονική περίοδο Τ διαιρείται σε ν χρονικά διαστήματα τ, έτσι ώστε σε κάθε χρονικό διάστημα Τ η ΚΜΕ να εξυπηρετεί μια από τις ν τερματικές μονάδες που είναι συνδεδεμένες στο σύστημα. Όταν μια τερματική μονάδα δεν ζητάει εξυπηρέτηση, ο χρόνος της διατίθεται για τις άλλες μονάδες. Το ίδιο συμβαίνει και όταν η εξυπηρέτηση μιας τερματικής μονάδας τελειώσει σε χρόνο μικρότερο από αυτόν που της διατίθεται. Αν αντίθετα ο χρόνος που διατίθεται σε μία τερματική μονάδα δεν επαρκεί η μονάδα αυτή θα εξυπηρετηθεί την επόμενη χρονική περίοδο Τ. Η τεχνική αυτή χρησιμοποιείται και για την ταυτόχρονη εκτέλεση περισσοτέρων εργασιών (προγραμμάτων).

### 6.1.5 Συστήματα multi user και multi tasking

Ένα λειτουργικό σύστημα λέγεται multi user όταν μπορεί να εξυπηρετεί ταυτόχρονα περισσότερους από ένα χρήστες. Στην αντίθετη περίπτωση το σύστημα λέγεται single user. Όταν ένα λειτουργικό σύστημα μπορεί να εκτελεί ταυτόχρονα περισσότερες από μία εργασίες (προγράμματα ή μέρη προγράμματος) λέγεται multi tasking, ενώ στην αντίθετη περίπτωση single tasking. Για παράδειγμα, το UNIX είναι λειτουργικό σύστημα multi user ενώ οι πρώτες εκδόσεις του MS-DOS ήταν single tasking - ωστόσο σε μεταγενέστερες εκδόσεις του προστέθηκαν κάποιες δυνατότητες multi tasking.

## 6.2 Διαχείριση της ΚΜΕ.

### 6.2.1 Η έννοια της διεργασίας (process)

Ένα ακολουθιακό πρόγραμμα είναι ένα πρόγραμμα που η εκτέλεση των εντολών του γίνεται ακολουθιακά (σειριακά). Η εκτέλεση του ονομάζεται διεργασία. Μια διεργασία είναι λοιπόν ένα πρόγραμμα ή τμήμα προγράμματος που εκτελείται σε κάποιον επεξεργαστή. Μια διεργασία μπορεί να περιγραφεί από το άνυσμα κατάστασης που περιλαμβάνει πληροφορίες

όπως, την τρέχουσα τιμή του μετρητή προγράμματος, τις τιμές των μεταβλητών, των καταχωρητών κ.λπ.

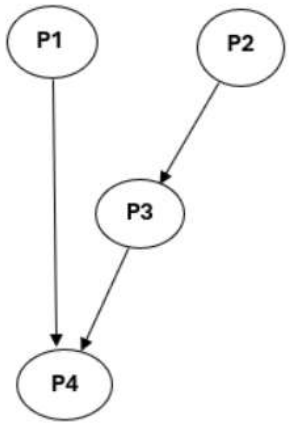
Η έννοια της διεργασίας βοηθάει στη μελέτη της οργάνωσης των λειτουργικών συστημάτων και τον τρόπο λειτουργίας τους. Ένα ταυτόχρονο (concurrent) πρόγραμμα καθορίζει δύο ή περισσότερα ακολουθιακά προγράμματα που μπορούν να εκτελούνται ταυτόχρονα σαν παράλληλες διεργασίες. Κατά την εκτέλεση ενός ταυτόχρονου προγράμματος έχουμε πολυπρογραμματισμό.

Ο γράφος προήγησης είναι ένας σχηματικός τρόπος για την παράσταση ταυτόχρονων προγραμμάτων. Ο γράφος προήγησης αποτελείται από κόμβους και κλάδους. Οι κόμβοι αντιστοιχούν σε διεργασίες, ενώ κλάδος από τον κόμβο i στον κόμβο j δηλώνει οτι η διεργασία Pj μπορεί να εκτελεστεί μόνο μετά την ολοκλήρωση της διεργασίας $P_i$. Έστω π.χ. οτι θέλουμε να υπολογίσουμε την αριθμητική έκφραση $x_1 * x_2 - (x_3 - x_4) / x_5$. Οι εκφράσεις $x_1 * x_2$ και $x_3 - x_4$ μπορούν να υπολογιστούν ταυτόχρονα, δεδομένου ότι είναι ανεξάρτητες. Αν παραστήσουμε με $P_1$ ως $P_4$ τις διεργασίες με τις οποίες υπολογίζονται οι παρακάτω εκφράσεις

$P_1 : t_1 = x_1 * x_2$ $P_2 : t_2 = x_3 - x_4$ $P_3 : t_3 = t_2 / x_5$

$P_4 : t_4 = t_1 - t_3$

ο γράφος προήγησης για την εκτέλεση των διεργασιών αυτών φαίνεται στο παρακάτω σχήμα:




Για την παράσταση ενός γράφου προήγησης, δηλαδή γενικά την παράσταση προβλημάτων με παράλληλες διεργασίες, εφαρμόζονται διάφορες μέθοδοι. Πιο γνωστή είναι οι σημαφορείς (semaphores). Ο σημαφορέας είναι μια ακέραιη μη αρνητική μεταβλητή, η οποία λαμβάνει ένα εύρος τιμών, υποδεικνύοντας με τον τρόπο αυτό τη διαθεσιμότητα πολλών μονάδων του ιδίου πόρου, πχ. εκτυπωτές ή μονάδες μνήμης. Υπάρχουν 2 εντολές που επενεργούν στο

σημαφορέα, οι οποίες εκτελούνται από τις διεργασίες που τρέχουν. Η εντολή UP προσθέτει μια μονάδα στο σημαφορέα ενώ η DOWN αφαιρεί μία μονάδα από τον σημαφορέα.

### 6.2.2 Χρονοδρομολόγηση της ΚΜΕ

Χρονοδρομολόγηση της ΚΜΕ είναι η ανάθεση του επεξεργαστή ή των επεξεργαστών ενός υπολογιστικού συστήματος στις διάφορες διεργασίες, και είναι από τα κυριότερα προβλήματα που αντιμετωπίζει το λειτουργικό σύστημα.

Σε ένα σύστημα πολυπρογραμματισμού έχουμε την ίδια στιγμή έναν αριθμό ανεξαρτήτων προγραμμάτων (διεργασιών) στην κεντρική μνήμη. Η ΚΜΕ μπορεί να μοιραστεί στις διάφορες διεργασίες βελτιώνοντας έτσι τη συνολική απόδοση του υπολογιστή. Η βασική ιδέα είναι οτι μια εργασία εκτελείται στην ΚΜΕ μέχρι να χρειαστεί να περιμένει για την εκτέλεση κάποιας λειτουργίας εισόδου/εξόδου. Επειδή οι ταχύτητες των μονάδων εισόδου/εξόδου είναι πολύ χαμηλότερες από την ταχύτητα της ΚΜΕ, οι λειτουργίες εισόδου/εξόδου είναι αργές. Σε αυτό το χρονικό διάστημα το λειτουργικό σύστημα απομακρύνει την εν λόγω διεργασία (εισόδου/εξόδου) και παραχωρεί την ΚΜΕ σε κάποια άλλη διεργασία. Έτσι βελτιώνεται η αποδοτικότητα του συστήματος με τη χρονική επικάλυψη των λειτουργιών εισόδου/εξόδου και εκείνων που απαιτούν εξυπηρέτηση από την ΚΜΕ.

Είδαμε ότι διεργασία είναι ένα πρόγραμμα σε εκτέλεση. Διεργασίες είναι εργασίες που δίνονται από τον χρήστη ή προέρχονται από το λειτουργικό σύστημα και εξυπηρετούν ειδικούς σκοπούς (αιτήσεις προγραμμάτων, λάθη κ.λ.π.) Η εκτέλεση μιας εργασίας είναι ένας κύκλος από εκτέλεση στην ΚΜΕ και αναμονής για λειτουργία εισόδου/εξόδου. Έτσι λέμε ότι έχουμε μία έκρηξη ΚΜΕ που ακολουθείται από έκρηξη εισόδου/εξόδου και ούτω καθεξής. Οι εκρήξεις αυτές διαφέρουν ανάλογα με τις διεργασίες και τον υπολογιστή, ανά γενικά υπάρχει μεγάλος αριθμός από μικρές σύντομες εκρήξεις και μικρός αριθμός από πολύ μεγάλες εκρήξεις. Όταν ένα πρόγραμμα έχει πολύ σύντομες εκρήξεις ΚΜΕ λέμε ότι είναι περιορισμένο από είσοδο/έξοδο, ενώ όταν έχει πολύ μεγάλες εκρήξεις ΚΜΕ λέμε ότι είναι περιορισμένο από την ΚΜΕ.

Μια διεργασία μπορεί να βρίσκεται σε διάφορες καταστάσεις. Οι σπουδαιότερες είναι οι παρακάτω. Όταν εκτελείται από την ΚΜΕ λέμε ότι είναι "εκτελούμενη". Όταν είναι έτοιμη να εκτελεστεί μόλις ελευθερωθεί η ΚΜΕ λέμε ότι είναι "έτοιμη" προς εκτέλεση. Η εργασία αυτή θα βρίσκεται ήδη στην Κεντρική Μνήμη. Όταν μια διεργασία έχει διακόψει την εκτέλεσή της περιμένοντας κάποιο γεγονός προκειμένου να συνεχίσει, λέμε ότι βρίσκεται σε "αναμονή". Μια νεοεισερχόμενη διαδικασία βρίσκεται στην κατάσταση "νέα". Προκειμένου να μεταβεί στην κατάσταση "έτοιμη" θα πρέπει να της διατεθούν η απαιτούμενη ΚΜ, μονάδες εισόδου/εξόδου κλπ. Όταν η εκτέλεση μιας διεργασίας τελειώσει, λέμε ότι είναι "ολοκληρωμένη".

Κάθε διεργασία παριστάνεται στο λειτουργικό σύστημα σαν ένα Σύνολο Ελέγχου Διεργασίας (ΣΕΔ ή Process Control Block – PCB). Το ΣΕΔ είναι ένα σύνολο δεδομένων που αφορούν την εν λόγω διεργασία και περιλαμβάνει : ένα δείκτη, (δηλαδή τη διεύθυνση κάποιου άλλου ΣΕΔ), έναν κωδικό αριθμό της διεργασίας, την κατάσταση της διεργασίας, τις διευθύνσεις μνήμης που χρησιμοποιεί η διεργασία, τα περιεχόμενα των καταχωρητών της ΚΜΕ όταν η διεργασία διακόπτεται, την προτεραιότητα της διεργασίας, καθώς και μερικές άλλες πληροφορίες, που αφορούν π.χ. αρχεία ή μονάδες εισόδου/εξόδου που χρησιμοποιεί η διεργασία.

Όπως είδαμε, έχουμε μεταγωγή της ΚΜΕ ανάμεσα σε διάφορες διεργασίες. Προκειμένου να γίνει μεταγωγή από μια διεργασία σε άλλη, το λειτουργικό σύστημα αποθηκεύει τις τιμές των καταχωρητών της διακοπτόμενης διεργασίας και φορτώνει τις τιμές των καταχωρητών της προς εκτέλεση διεργασίας. Σε κάθε στιγμή εκτελείται μία μόνο διεργασία, ενώ οι υπόλοιπες περιμένουν να ελευθερωθεί η ΚΜΕ. Οι διεργασίες αυτές (που θα βρίσκονται στην κατάσταση “έτοιμη”), τοποθετούνται σε μια λίστα που λέγεται Λίστα Έτοιμων Διεργασιών (RQ - Ready Queue). Η RQ είναι μια συνδεδεμένη λίστα (θα δούμε τι είναι μια συνδεδεμένη λίστα στο Κεφάλαιο 8).

Ένα λειτουργικό σύστημα λειτουργεί με πολλά επίπεδα χρονοδρομολόγησης. Ο μακροχρόνιος δρομολογητής καθορίζει ποιες διαδικασίες θα εισέλθουν στο σύστημα για επεξεργασία. Συνήθως υποβάλλονται περισσότερες εργασίες από όσες μπορεί να εκτελεστούν αμέσως και οδηγούνται σε κάποια μονάδα Βοηθητικής Μνήμης. Ο μακροχρόνιος δρομολογητής επιλέγει ορισμένες από αυτό το σύνολο και τις οδηγεί στην Κεντρική Μνήμη. Ο βραχυχρόνιος δρομολογητής επιλέγει από τις διεργασίες που βρίσκονται στην Κεντρική Μνήμη και παραχωρεί κάποια από αυτές στην ΚΜΕ. Ο βραχυχρόνιος δρομολογητής χρειάζεται να κάνει επιλογή πιο συχνά από τον μακροχρόνιο δρομολογητή. Είναι πολύ σημαντικό ο μακροχρόνιος δρομολογητής να κάνει κάνει καλή επιλογή ώστε να στέλνει στην RQ ένα ισοζυγισμένο μείγμα από διεργασίες περιορισμένων από την ΚΜΕ και περιορισμένων από είσοδο/έξοδο. Μερικά συστήματα έχουν και τρίτο επίπεδο χρονοδρομολόγησης, τον μεσοπρόθεσμο δρομολογητή, που μετακινεί προσωρινά οι εργασίες από την κύρια μνήμη στην βοηθητική μνήμη.

### 6.2.3 Αλγόριθμοι χρονοδρομολόγησης

Για να αποφασιστεί ποια από τις διεργασίες της RQ θα παραχωρηθεί στην ΚΜΕ, χρησιμοποιούνται διάφορες μέθοδοι. Θα δούμε παρακάτω τις βασικούς αλγορίθμους που εφαρμόζουμε, αφού πρώτα δούμε τα κυριότερα κριτήρια επιλογής :

α) Η χρησιμοποίηση της ΚΜΕ που μπορεί να φτάνει ως και 90%.

β) Η ρυθμοαπόδοση, δηλαδή ο αριθμός των διεργασιών που εκτελούνται στη μονάδα του χρόνου

γ) Ο χρόνος εξυπηρέτησης (turnaround time), δηλαδή ο συνολικός χρόνος από τη στιγμή υποβολής μιας διεργασίας μέχρι τη στιγμή ολοκλήρωσης της

δ) Ο χρόνος αναμονής μίας διεργασίας στην RQ

ε) Ο χρόνος απόκρισης μιας διεργασίας, δηλαδή ο χρόνος από την υποβολή μιας διεργασίας μέχρι την έναρξη της απόκρισης. Στο κριτήριο αυτό δεν λαμβάνουμε υπόψη το χρόνο που απαιτείται για τη λειτουργία εξόδου της απόκρισης.

Ας δούμε τη συνέχεια τους σπουδαιότερους αλγόριθμους.

1) FCFS (First Come First Served)

Οι διεργασίες εκτελούνται με βάση τη σειρά άφιξής τους. Η υλοποίηση του αλγορίθμου αυτού γίνεται με μια ουρά FIFO (First In First Out). Η απόδοση του αλγορίθμου αυτού είναι χαμηλή.

2) SJF (Shortest Job First)

Στον αλγόριθμο αυτό, σε κάθε διεργασία αντιστοιχίζεται το μήκος της επόμενης έκρηξης ΚΜΕ και ή ΚΜΕ παραχωρείται στη διεργασία με τη μικρότερη έκρηξη ΚΜΕ. Ο αλγόριθμος αυτός έχει καλή απόδοση, αλλά η πραγματική δυσκολία βρίσκεται στην πρόβλεψη της διαρκείας της επόμενης έκρηξης ΚΜΕ κάθε διεργασίας. Γι' αυτό, ενώ χρησιμοποιείται αποτελεσματικά στο επίπεδο μακροχρόνιας δρομολόγησης, όπου οι διεργασίες δίνονται με σύστημα ομάδας, στη βραχυχρόνια χρονοδρομολόγηση υλοποείται δύσκολα. Μια λύση είναι να εκτιμήσουμε τη διάρκεια της επόμενης έκρηξης ΚΜΕ με βάση τις προηγούμενες (δηλ. με στατιστικό τρόπο)

3) HRF (Highest-Response Ratio First)

Ο αλγόριθμος αυτός προσπαθεί να συνδυάσει τα πλεονεκτήματα των 2 προηγούμενων αλγόριθμων. Επιλέγει τη διεργασία με το μεγαλύτερο λόγο απόκρισης. Ο λόγος απόκρισης ορίζεται σαν το πηλίκο.

$$\frac{\text{Χρόνος απόκρισης}}{\text{Χρόνος εκτέλεσης.}}$$

Ο παράγοντας αυτός εκφράζει το ρυθμό επεξεργασίας του υπολογιστή, όπως τον αντιλαμβάνεται ο χρήστης. Ο αλγόριθμος αυτός ευνοεί τις μικρές διεργασίες, αλλά οριοθετεί και το χρόνο αναμονής των μεγαλύτερων.

4) Round Robin

Ο αλγόριθμος αυτός σχεδιάστηκε για συστήματα καταμερισμού χρόνου (time sharing). Σύμφωνα με τον αλγόριθμο αυτό, ορίζεται μια μικρή μονάδα χρόνου, που λέγεται quantum χρόνου (συνήθως είναι από 10 ως 100 msecond). Ο δρομολογητής παραχωρεί την ΚΜΕ σε κάθε διεργασία της RQ για χρονικό διάστημα το πολύ ίσο με το quantum χρόνου. Αν μία διεργασία απαιτεί έκρηξη ΚΜΕ μικρότερη από το quantum χρόνου, αποδεσμεύει από μόνη της την ΚΜΕ. Αντίθετα, αν η έκρηξη της ΚΜΕ είναι μεγαλύτερη από το quantum χρόνου, η διεργασία θα διακοπεί, τα περιεχόμενα των καταχωρητών της θα αποθηκευτούν στο ΣΕΔ της διεργασίας αυτής και η διεργασία θα μπει στο τέλος της RQ και θα περιμένει τον επόμενο κύκλο για να συνεχιστεί η εκτέλεση της.

Η απόδοση του αλγορίθμου αυτού εξαρτάται αποφασιστικά από το μέγεθος του quantum χρόνου. Αν αυτό είναι πολύ μικρό, θα έχουν χρονική επιβάρυνση του συστήματος για τη συνεχή αποθήκευση των καταχωρητών της διακοπείσας διεργασίας και φόρτωση των καταχωρητών της νέας. Αντίθετα, πολύ μεγάλο quantum χρόνου οδηγεί στην τεχνική FCFS.

Δρομολόγηση με προτεραιότητες

Στους αλγορίθμους που είδαμε παραπάνω η προτεραιότητα που θα δοθεί σε μία διεργασία καθορίζεται εσωτερικά, δηλαδή με βάση κάποιες μετρήσιμες ποσότητες της διεργασίας. Υπάρχουν όμως και εξωτερικές προτεραιότητες που εισάγονται εξωτερικά από το λειτουργικό σύστημα, και που έχουν να κάνουν με την ιδιότητα του χρήστη που θέτει τις διεργασίες, το μέγεθος της χρέωσης του υπολογιστή κ.λ.π.

Επίσης σε ορισμένους αλγόριθμους και υπό ορισμένες συνθήκες μπορεί να αναβάλλονται επ'αόριστον ορισμένες διεργασίες. Για το πρόβλημα αυτό εφαρμόζεται η τεχνική της ηλικίωσης, όπου σταδιακά αυξάνουμε την προτεραιότητα μιας διεργασίας ανάλογα με την αναμονή της στο σύστημα, ώστε κάποτε να αποκτήσει υψηλή προτεραιότητα και να εκτελεστεί.

### 6.3 Διαχείριση μνήμης

Μια άλλη βασική λειτουργία του λειτουργικού συστήματος είναι η διαχείριση της μνήμης του υπολογιστή. Ο χρήστης αντιλαμβάνεται τη μνήμη σαν ένα σύνολο συμβολικών διευθύνσεων και δεδομένων. Το λειτουργικό σύστημα αναλαμβάνει να κάνει την αντιστοίχιση από το χώρο των συμβολικών διευθύνσεων στο χώρο των απολύτων διευθύνσεων. Διακρίνουμε έτσι τη λογική, την ιδεατή και την φυσική μνήμη. Με την εισαγωγή του πολυπρογραμματισμού στα υπολογιστικά συστήματα αναπτύχθηκαν τεχνικές διαχείρισης της μνήμης από πολλές παράλληλες εργασίες. Με τις τεχνικές αυτές γίνεται η κατανομή της μνήμης στις διάφορες εργασίες, η φαινομενική της επαύξηση και εξασφαλίζεται η ανεξαρτησία των εργασιών χωρίς την παρέμβαση του χρήστη. Θα δούμε παρακάτω τις βασικές έννοιες και τεχνικές διαχείρισης μνήμης.

#### 6.3.1 Αμοιβαία ανταλλαγή (swapping)

Τα προγράμματα μέχρι να εκτελεστούν παραμένουν στη βοηθητική μνήμη (π.χ. δίσκος). Από εκεί καλείται ένα πρόγραμμα στην κεντρική μνήμη για να εκτελεστεί και όταν τελειώσει η εκτέλεση, το πρόγραμμα ξαναστέλνεται στη βοηθητική μνήμη και τη θέση του καταλαμβάνει κάποιο άλλο. Η διαδικασία αυτή λέγεται αμοιβαία ανταλλαγή και ο χρόνος της είναι μεγάλος σε σχέση με την ταχύτητα της ΚΜΕ. Βέβαια, φροντίζουμε ώστε στο χρόνο αυτό η ΚΜΕ να μην μένει αδρανής.

### 6.3.2 Overlays και ιδεατή μνήμη

Την πρώτη περίοδο που εμφανίστηκαν Η/Υ η μνήμη τους ήταν μικρή και ακριβή. Ο χρήστης έπρεπε να χρησιμοποιήσει βοηθητική μνήμη και χώριζε το πρόγραμμά του σε έναν αριθμό τμημάτων (overlays), καθένα από τα οποία χωρούσαν στην κεντρική μνήμη. Για να τρέξει το πρόγραμμα έπρεπε να φορτώσει στην κεντρική μνήμη το πρώτο overlay το οποίο αφού έτρεχε καλούσε ο επόμενο overlay κ.λ.π. Ο χρήστης ήταν υπεύθυνος για όλη τη διαδικασία διαχωρισμού και διαχείρισης των overlays. Στη συνέχεια βρέθηκε η κατάλληλη μέθοδος για να εκτελείται η διαδικασία αυτόματα από τον υπολογιστή. Η μέθοδος αυτή στηρίζεται στην τεχνική της ιδεατής μνήμης που είδαμε στο κεφάλαιο 5.

### 6.3.3 Σελιδοποίηση (paging)

Εξετάζοντας την ιδεατή μνήμη είδαμε την τεχνική της σελιδοποίησης, όπου ο χώρος διευθύνσεων (βοηθητική μνήμη) χωρίζεται σε σελίδες (pages) ενώ ο χώρος φυσικών διευθύνσεων (κεντρική μνήμη - ΚΜ) χωρίζεται σε ενότητες (frames) όπου το μέγεθος της σελίδας είναι ίσο με το μέγεθος της ενότητας. Με τη βοήθεια του πίνακα απεικόνισης μνήμης μια ιδεατή διεύθυνση του χώρου διευθύνσεων αντιστοιχίζεται σε μια φυσική διεύθυνση του χώρου μνήμης.

Με την τεχνική της σελιδοποίησης επιτρέπεται στη μνήμη που ανατίθεται σε ένα πρόγραμμα να μην είναι συνεχής αλλά διασκορπισμένη, δεδομένου ότι ενώ οι σελίδες είναι λογικά συνεχείς, οι ενότητες δεν είναι με τον ίδιο τρόπο συνεχείς.

Στο κεφάλαιο 5 είδαμε ένα απλό παράδειγμα συστήματος εικονικής μνήμης όπου ο χώρος διευθύνσεων είναι 8Κ = 8 σελίδες, ο χώρος μνήμης (η ΚΜ) είναι 4Κ = 4 ενότητες και επομένως κάθε στιγμή μέχρι 4 σελίδες μπορούν να βρίσκονται στην.ΚΜ σε οποιαδήποτε από τις 4 ενότητες της.

Είδαμε επίσης ότι αν κατά την αναζήτηση μιας σελίδας στον πίνακα απεικόνισης μνήμης το bit παρουσίας είναι μηδέν (που σημαίνει ότι αντίστοιχη σελίδα δε βρίσκεται στην ΚΜ) ενεργοποιείται μέσω του λειτουργικού συστήματος διαδικασία μεταφοράς της ζητούμενης σελίδας από τη βοηθητική στην κεντρική μνήμη.

<u>Αντικατάσταση σελίδας</u>

Σε αυτήν την περίπτωση λοιπόν, το λειτουργικό σύστημα πρέπει να αποφασίζει :

α) Ποιά σελίδα στην ΚΜ πρέπει να αφαιρεθεί για να δημιουργήσει χώρο για τη νέα σελίδα

β) Στην γενική περίπτωση, πότε μια νέα σελίδα μεταφέρεται από την βοηθητική στην ΚΜ

γ) Πού θα τοποθετηθεί μέσα στην ΚΜ η σελίδα

Όταν αρχίσει η εκτέλεση ενός προγράμματος, μία ή περισσότερες σελίδες μεταφέρονται μέσα στην.ΚΜ και ο πίνακας απεικόνισης μνήμης (ή πίνακας σελίδων) παίρνει τις κατάλληλες τιμές ώστε να δείχνει τις θέσεις των σελίδων στην ΚΜ. Ένα πρόγραμμα εκτελείται από την ΚΜ μέχρι να ζητηθεί προσπέλαση σε σελίδα που δεν έχει μεταφερθεί στην ΚΜ. Η συνθήκη

αυτή λέγεται λάθος σελίδας (page fault). Τότε διακόπτεται η εκτέλεση του τρέχοντος προγράμματος και ο έλεγχος περνάει στο λειτουργικό σύστημα που αναλαμβάνει να εκτελέσει τη μεταφορά της ζητούμενης σελίδας στην ΚΜ (πρόκειται δηλαδή για λειτουργία.εισόδου/εξόδου). Όταν η διαδικασία αυτή ολοκληρωθεί, το πρόγραμμα που είχε διακοπεί, μπορεί να συνεχίσει την εκτέλεση του.

Κατά τη διαδικασία μεταφοράς σελίδας προς την ΚΜ, αν η ΚΜ είναι γεμάτη, θα πρέπει να αφαιρεθεί από αυτή μια σελίδα για να δημιουργηθεί ο αιτούμενος χώρος. Για να γίνει η επιλογή της σελίδας αυτής, υπάρχουν διάφοροι αλγόριθμοι, εκ των οποίων οι πιο συνηθισμένοι είναι :

Να επιλεγεί η σελίδα που παρέμεινε στην ΚΜ πιο πολύ χρόνο. Κάθε φορά που φορτώνεται η σελίδα στην ΚΜ, ο κωδικός της μπαίνει σε μια ουρά FIFO (First-In- First Out). Όταν πρέπει να αφαιρεθεί κάποια σελίδα, επιλέγεται εκείνη που μπήκε πιο παλιά, της οποίας ο κωδικός θα βρίσκεται στην αρχή της ουράς FIFO. Ο αλγόριθμος αυτός υλοποιείται εύκολα, αλλά παρουσιάζει κάποια προβλήματα.

Ο δεύτερος αλγόριθμος επιλέγει τη λιγότερο πρόσφατα χρησιμοποιημένη σελίδα (least recently used : LRU). Για τη μέθοδο αυτή απαιτείται για κάθε σελίδα που βρίσκεται στην ΚΜ ένας μετρητής. Όταν γίνεται προσπέλαση σε μια σελίδα, ο αντίστοιχος μετρητής μηδενίζεται, ενώ σε τακτά χρονικά διαστήματα όλοι οι μετρητές αυξάνονται κατά 1. Έτσι όταν πρόκειται να αντικατασταθεί μια σελίδα, επιλέγεται εκείνη με το μεγαλύτερο μετρητή.

### 6.3.4. Μέγεθος σελίδας και κατακερματισμός (fragmentation)

Αν τα προγράμματα και δεδομένα του χρήστη τυχαίνει να γεμίζουν ακέραιο αριθμό σελίδων ακριβώς τότε δεν έχουμε καθόλου σπατάλη χώρου. Αν όμως δεν γεμίζουν ακέραιο αριθμό σελίδων, θα υπάρχει στην τελευταία σελίδα κάποιος μη χρησιμοποιούμενος χώρος. ‘Οποτε η σελίδα αυτή βρίσκεται στην ΚΜ, το μη χρησιμοποιούμενο μέρος της καταλαμβάνει χώρο στην ΚΜ χωρίς να είναι χρήσιμο. Έτσι δημιουργούνται ‘τρύπες’, δηλαδή σπατάλη χρήσιμου χώρου μνήμης. Το πρόβλημα αυτό ονομάζεται κατακερματισμός (fragmentation).

Αν το μέγεθος της σελίδας είναι ν λέξεις, ο μέσος όρος σπαταλούμενου χώρου στην τελευταία σελίδα θα είναι (στατιστικά) ν/2 λέξεις. Αυτό συνηγορεί καταρχήν στη χρήση μικρής σελίδας. Όμως αυτό σημαίνει πολλές σελίδες και μεγάλο πίνακα σελίδων. Επιπλέον, θα έχουμε περισσότερες αντικαταστάσεις σελίδων, που σημαίνει και σπατάλη χρόνου. Γι’ αυτό η επιλογή του μεγέθους σελίδας είναι θέμα που απαιτεί μελέτη.

### 6.3.5 Κατάτμηση (segmentation)

Με τη σελιδοποίηση είδαμε μια μέθοδο μέσω της οποίας ο χρήστης δεν ‘βλέπει’ απευθείας τη φυσική μνήμη του υπολογιστή αλλά την ιδεατή μνήμη. Είναι είναι επίσης χρήσιμο και αποδοτικό για τον προγραμματιστή να βλέπει την μνήμη όχι σαν ένα σύνολο από δεδομένα

και εντολές, αλλά σαν μια συλλογή από κομμάτια μεταβλητού μήκους, καθένα από τα οποία περιλαμβάνει ομοειδή πράγματα (υπορουτίνες, συναρτήσεις, πίνακες κ.λ.π). Ο χρήστης αναφέρεται σε ένα από αυτά τα κομμάτια με ένα όνομα, χωρίς να ενδιαφέρεται πού ακριβώς στη μνήμη βρίσκεται. Η μέθοδος αυτή λέγεται κατάτμηση (segmentation).

Το σύνολο των τμημάτων αυτών (segments) αποτελεί το χώρο των λογικών διευθύνσεων. Κάθε τμήμα έχει κάποιο όνομα και κάποιο μήκος.

Όταν ένα πρόγραμμα διοχετεύεται στο μεταφραστή (compiler) μιας γλώσσας, τότε ο μεταφραστής δημιουργεί αυτόματα τα διάφορα τμήματα. Για παράδειγμα θα δημιουργήσει διαφορετικά τμήματα (segments) για :

α) τις global (καθολικές) μεταβλητές

β) τη στοίβα (stack) με τις κλήσεις των υπορουτινών

γ) τις διάφορες υπορουτίνες και συναρτήσεις

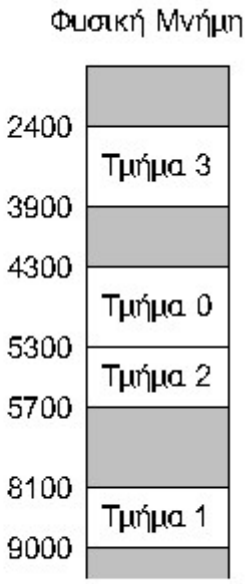
δ) τις τοπικές (local) μεταβλητές για κάθε υπορουτίνα και συνάρτηση.

Για την υλοποίηση της μεθόδου αυτής απαιτείται ένας πίνακας κατάτμησης (segment table) στον οποίον για κάθε τμήμα καταχωρούνται η βάση (base), που δείχνει τη διεύθυνση θέσης μνήμης που αρχίζει το τμήμα, και το όριο (limit), που δείχνει το μήκος του τμήματος.

Στο παράδειγμα που φαίνεται στο παρακάτω σχήμα έχουμε 4 τμήματα (τα υπ’ αριθμόν 0, 1, 2, 3). Ο πίνακας κατάτμησης είναι ένας δισδιάστατος πίνακας, όπου το πρώτο στοιχείο μας δίνει το όριο (σε λέξεις) και το δεύτερο τη βάση κάθε τμήματος. Για παράδειγμα το τμήμα 2 έχει μήκος 400 λέξεις, ξεκινώντας από τη θέση μνήμης 5300. Άν θέλουμε να προσπελάσουμε την 57η θέση μνήμης του τμήματος 2, αυτή θα είναι στην πραγματικότητα η θέση μνήμης με διεύθυνση 5300 + 57 = 5357. Αν όμως θελήσουμε να προσπελάσουμε την 500ή θέση μνήμης του τμήματος 2, τότε το Λειτουργικό Σύστημα θα ανιχνεύσει το λάθος (κάνοντας σύγκριση με το όριο) και δεν θα επιτρέψει προσπέλαση στη μνήμη.

Πίνακας Κατάτμησης

| Τμήμα | Όριο | Βάση |
|---|---|---|
| 0 | 1000 | 4300 |
| 1 | 900 | 8100 |
| 2 | 400 | 5300 |
| 3 | 1500 | 2400 |




Παραβάλλοντας τις τεχνικές της σελιδοποίησης και της κατάτμησης, σημειώνουμε ότι τα περιεχόμενα μιας σελίδας είναι τυχαία, ενώ τα περιεχόμενα ενός τμήματος είναι ομοειδή (π.χ. υπορουτίνες). Ακόμη, ενώ ο χρήστης δεν αντιλαμβάνεται τη λειτουργία της σελιδοποίησης, τη λειτουργία της κατάτμησης την αντιλαμβάνεται, καθώς και το τι καταχωρείται σε κάθε τμήμα.

## 6.4 Διαχείριση αρχείων

### 6.4.1 Αρχεία και μέσα αποθήκευσης

Η δυνατότητα να αποθηκεύσουμε πληροφορίες στην Κεντρική Μνήμη του υπολογιστή αλλά κυρίως στη βοηθητική του μνήμη, είναι ένα από τα σημαντικότερα χαρακτηριστικά του υπολογιστή, που του δίνει τεράστιες δυνατότητες.

Αρχείο (file) λέγεται γενικά ένα σύνολο πληροφοριών, που καταχωρούνται συνήθως σε κάποιο μαγνητικό μέσο, σύμφωνα με κάποια φυσική και λογική οργάνωση. Ο τρόπος οργάνωσης των αρχείων έχει σχέση με το μέσο αποθήκευσης. Με τη χρήση των μαγνητικών δίσκων έγινε δυνατή η άμεση προσπέλαση σε κάθε σημείο του δίσκου, άρα και σε κάθε αρχείο (ενώ στην μαγνητική ταινία, που χρησιμοποιείτο παλαιότερα, ο τρόπος οργάνωσης ήταν σειριακός, δηλαδή το ένα αρχείο μετά το άλλο). Στο Κεφάλαιο 5 είδαμε τη φυσική οργάνωση του δίσκου που διαιρείται σε ίχνη και τομείς. Η άμεση προσπέλαση άλλαξε τη φιλοσοφία του τρόπου οργάνωσης αρχείων και έτσι εισήχθη η έννοια του καταλόγου αρχείων (catalog directory). Όπως επίσης είδαμε στο Κεφάλαιο 5, η εγγραφή προς και η ανάγνωση από τη μνήμη γίνεται με ολόκληρα εγγραφήματα, που είναι σύνολα λέξεων. Η διαδικασία αυτή γίνεται μέσω της ΚΜΕ με εντολές εισόδου-εξόδου (Read-Write).

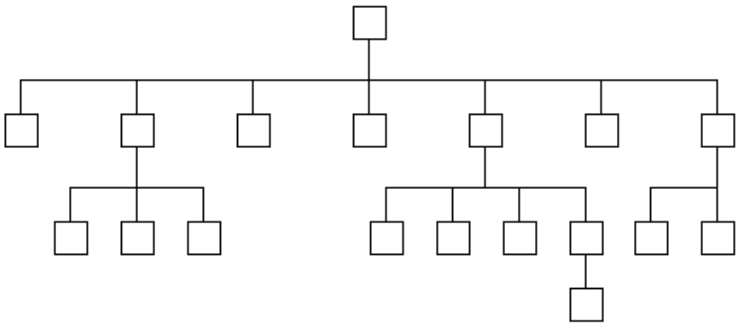
### 6.4.2 Κατάλογοι Αρχείων

Η διαχείριση των αρχείων είναι και αυτή μία από τις βασικές εργασίες του Λειτουργικού Συστήματος. Ένας κατάλογος (catalog ή directory) ενός δίσκου περιλαμβάνει για κάθε αρχείο του δίσκου το όνομά του και κάποια άλλα χρήσιμα στοιχεία και όταν γίνεται αναφορά σε κάποιο αρχείο, στέλνει την κεφαλή του δίσκου στην κατάλληλη διεύθυνση. Εκτός από το όνομα αρχείου, άλλα χρήσιμα στοιχεία που αναφέρονται είναι το μήκος του αρχείου, το είδος του (π.χ. πρόγραμμα σε κάποια γλώσσα προγραμματισμού) η ημερομηνία που δημιουργήθηκε, πότε έγινε η τελευταία προσπέλαση σε αυτό, πότε έγινε η τελευταία αλλαγή, σύνολο προσπελάσεων, πληροφορίες για το πού βρίσκεται το αρχείο αυτό στο δίσκο (αριθμοί ιχνών και τομέων) κ.λ.π.

Για την προστασία των αρχείων υπάρχουν διάφορες μέθοδοι. Μία από αυτές είναι ο “ιδιοκτήτης” του αρχείου να ορίσει ένα κρυφό κωδικό (password) ο οποίος αν δεν δοθεί σωστά δεν θα επιτρέπεται η προσπέλαση του αρχείου. Άλλος τρόπος είναι ο ιδιοκτήτης του αρχείου να παρέχει μία λίστα με τους χρήστες στους οποίους τα προγράμματα μπορούν να προσπελάσουν το αρχείο αυτό.

### 6.4.3 Ιεραρχική δομή αρχείων

Η αναζήτηση ενός αρχείου μέσα από έναν κατάλογο, μπορεί να είναι δύσκολη και χρονοβόρα διαδικασία όταν έχουμε μεγάλο πλήθος αρχείων (οι δίσκοι σήμερα έχουν τεράστια χωρητικότητα). Η λύση για αυτό το πρόβλημα είναι να μοιράσουμε τα αρχεία σε ομάδες και να τα εντάξουμε σε καταλόγους με μία ιεραρχική δομή όπως στο παρακάτω σχήμα :



Ο βασικός κατάλογος λέγεται ρίζα (root) και μπορεί να περιλαμβάνει αρχεία αλλά και υποκαταλόγους (subdirectories) καθένας από τους οποίους περιλαμβάνει αρχεία αλλά και υποκαταλόγους κ.ο.κ. Έτσι δημιουργείται ένα αντεστραμμένο δέντρο από υποκαταλόγους, Τα “φύλλα” αυτού του δέντρου θα είναι τα αρχεία.

Η δομή αυτή με τον κατάλογο και τους υποκαταλόγους είναι ιδιαίτερα χρήσιμη, π.χ. σε περιπτώσεις που κάποιος χρήστης δουλεύει για διαφορετικά έργα στον ίδιο υπολογιστή (οπότε μπορεί να έχει και ξεχωριστούς υποκαταλόγους για κάθε έργο) ή δουλεύουν πολλοί χρήστες στον ίδιο υπολογιστή, οπότε ο κάθε χρήστης έχει δικό του υποκατάλογο.

### 6.4.4 Εντολές αρχείων

Ένα Λειτουργικό Σύστημα θα περιλαμβάνει οπωσδήποτε εντολές αρχείων για τις ακόλουθες λειτουργίες:

- Δημιουργία αρχείου και εισαγωγή του στον κατάλογο του “ιδιοκτήτη”
- Διαγραφή αρχείου από τον κατάλογο
- Αλλαγή ονομασίας αρχείου
- Μεταβολή του καθεστώτος προστασίας (protection status) ενός αρχείου

Χαρακτηριστικό είναι οτι πολλά Λειτουργικά Συστήματα (όπως το παλαιότερο MSDOS , το UNIX και το LINUX) αντιμετωπίζουν τις περιφερειακές συσκευές σαν αρχεία και επομένως μπορούμε να χρησιμοποιούμε εντολές αρχείων με όρισμα το συμβολικό όνομα μιας περιφερειακής συσκευής.

# ΚΕΦΑΛΑΙΟ 7. ΑΛΓΟΡΙΘΜΟΙ – ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ – ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

## 7.1 Προγράμματα και προγραμματισμός

Όπως είδαμε και στο Κεφάλαιο 4 για να επιλύσουμε ένα πρόβλημα στον υπολογιστή πρέπει να εισάγουμε το κατάλληλο πρόγραμμα με τα αντίστοιχα δεδομένα, και το πρόγραμμα αυτό είναι μία σειρά εντολών γραμμένες σε κάποια γλώσσα προγραμματισμού. Τα αποτελέσματα που θα πάρουμε μετά την εκτέλεση του προγράμματος είναι και η λύση του προβλήματος. Η όλη διαδικασία λέγεται προγραμματισμός. Η όλη διαδικασία της δημιουργίας ενός προγράμματος αναλύεται σε βήματα τα σπουδαιότερα από τα οποία είναι :

α) Ανάλυση και κατάλληλη περιγραφή του προβλήματος

β) Επίλυση του προβλήματος με τη διατύπωση ενός κατάλληλου αλγόριθμου

γ) Σχηματική παράσταση του αλγορίθμου με ένα λογικό διάγραμμα ροής (flow chart)

δ) Μεταφορά του αλγόριθμου σε κάποια γλώσσα προγραμματισμού και διαμόρφωση του προγράμματος

ε) Εκτέλεση του προγράμματος του υπολογιστή, με παράλληλο έλεγχο για ύπαρξη συντακτικών και ορθογραφικών σφαλμάτων στο πρόγραμμα

στ) Έλεγχος αποτελεσμάτων έτσι ώστε να διαπιστώσουμε ότι το πρόγραμμα λειτουργεί σωστά με οποιαδήποτε δεδομένα

ζ) Διόρθωση σφαλμάτων και επανέλεγχος

η) Τεκμηρίωση του προγράμματος (documentation)

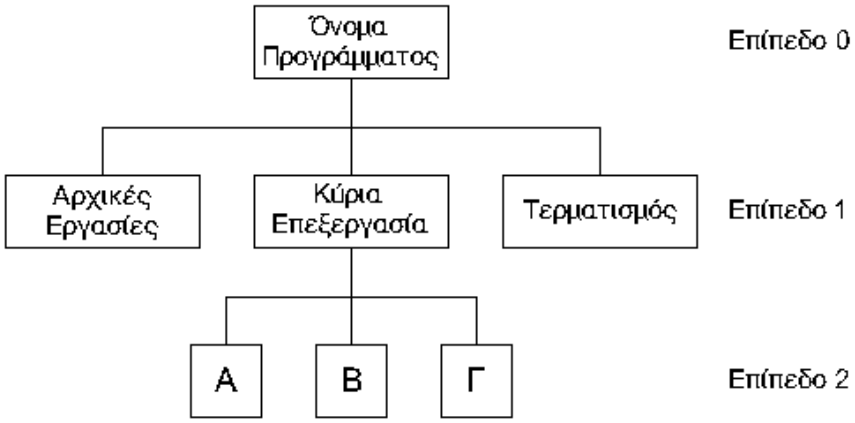
Πρέπει εδώ να αναφερθεί ότι ο προγραμματισμός είναι μία δύσκολη διαδικασία που απαιτεί πείρα και οξυδέρκεια εκ μέρους του προγραμματιστή, καθώς και καλή γνώση της γλώσσας προγραμματισμού που χρησιμοποιεί. Είναι επίσης σημαντικό να φτιαχτεί όχι οποιοδήποτε πρόγραμμα που να επιλύει ένα πρόβλημα, αλλά το πρόγραμμα που θα φτιαχτεί να είναι όσο το δυνατόν καλλίτερο. Προσοχή πρέπει να δοθεί βέβαια σε όλα τα παραπάνω βήματα του προγραμματισμού. Η σωστή και ξεκάθαρη διατύπωση του προβλήματος (μαθηματική ή λογική διατύπωση) σίγουρα δίνει τις κατευθύνσεις για τη λύση του. Η διατύπωση του σωστού αλγόριθμου στη συνέχεια είναι η βάση για το γράψιμο του βέλτιστου προγράμματος. Η επιλογή της γλώσσας προγραμματισμού παίζει επίσης πολύ σημαντικό ρόλο. Δεν είναι σκόπιμο π.χ. να χρησιμοποιήσουμε μία γλώσσα κατάλληλη για επιστημονικά προβλήματα για την ανάπτυξη μιας εμπορικής εφαρμογής, ακόμη και αν αυτό είναι δυνατόν. Κατά την επιλογή της γλώσσας πρέπει επίσης να σκεφτούμε αν η ταχύτητα σχεδίασης, μετάφρασης και εκτέλεσης του προγράμματος που προκύπτει είναι παραδεκτή. Ο έλεγχος του προγράμματος πρέπει να είναι διεξοδικός για να καλύψει όλα τα ενδεχόμενα σφάλματος , ενώ και η τεκμηρίωση του προγράμματος είναι σημαντική εργασία.

### 7.1.1 Τεχνικές προγραμματισμού

α) Ιεραρχική σχεδίαση (Τεχνική Top-Down)

Η τεχνική αυτή εισάγει μία ιεραρχικότητα στο σχεδιασμό και την ανάπτυξη των προγραμμάτων (ιεραρχική σχεδίαση προγράμματος). Σύμφωνα με την τεχνική αυτή καθορίζουμε πρώτα τις βασικές λειτουργίες του προβλήματος και με βάση αυτές, καθορίζουμε τις υπολειτουργίες που προκύπτουν. Με τον τρόπο αυτό έχουμε λειτουργικά επίπεδα (levels), που τα αριθμούμε με τη σειρά 0, 1, 2, …

Στο παρακάτω σχήμα βλέπουμε ένα ιεραρχικό διάγραμμα λειτουργιών για ένα γενικό πρόγραμμα επεξεργασίας κάποιων δεδομένων :




Η τεχνική top-down παρέχει αρκετά πλεονεκτήματα, τόσο στην κωδικοποίηση του προγράμματος (δηλαδή το γράψιμο των εντολών) όσο και στον έλεγχο (test) του προγράμματος.

β) Τμηματικός ή αρθρωτός προγραμματισμός (modular programming)

Με την τεχνική αυτή τα προγράμματα χωρίζονται σε modules (τμήματα ή ενότητες). Κάθε module είναι ένα σύνολο εντολών που εκτελούν κάποια συγκεκριμένη λειτουργία, έχει τις δικές του μεταβλητές, ενώ είναι λιγότερο ή περισσότερο ανεξάρτητο από τα άλλα μέρη του προγράμματος. Η επικοινωνία του με αυτά γίνεται μέσω ορισμένων αυστηρά καθορισμένων μεταβλητών.

γ) Δομημένος προγραμματισμός (structured programming)

Σύμφωνα με την τεχνική αυτή ένα πρόγραμμα θα πρέπει να φτιαχτεί με βάση τους εξής κανόνες

1. Απλή ακολουθία εντολών
2 Ελεγχόμενη επανάληψη ακολουθία εντολών (εντολές της μορφής DO - WHILE )
3. Εντολή της μορφής IF-THEN-ELSE. Μετά το THEN (συνθήκη αληθής) ή το ELSE (συνθήκη ψευδής) ακολουθούν ομάδες εντολών.

Κάθε ομάδα μπορεί να περιλαμβάνει οποιονδήποτε από τους 3 κανόνες. Χαρακτηριστικό της τεχνικής αυτής είναι ο περιορισμός ή αποκλεισμός της χρήσης της εντολής GOTO.

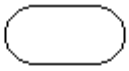
δ) Αντικειμενοστραφής προγραμματισμός (object oriented programming). Σύμφωνα με την τεχνική αυτή, δημιουργούνται κατ'αρχήν κλάσεις (classes), οι οποίες είναι μορφότυπα (templates) για τη δημιουργία αντικειμένων (objects). Κάθε αντικείμενο, που είναι "στιγμιότυπο" κάποιας κλάσης, είναι μία οργανωμένη ενότητα κώδικα που περιέχει δεδομένα (attributes) και λειτουργίες ή μεθόδους (methods) οι οποίες δείχνουν πώς μία κλάση ενεργεί και επικοινωνεί με άλλες κλάσεις. Κάθε αντικείμενο αποτελεί δηλαδή ξεχωριστή οντότητα που ενσωματώνει τα δεδομένα και τις μεθόδους της. Η δυνατότητα αυτή ενός αντικειμένου να συνδυάζει εσωτερικά τα δεδομένα και τις μεθόδους του ονομάζεται ενθυλάκωση (encapsulation). Χαρακτηριστικά της τεχνικής αυτής είναι οτι οι κλάσεις μπορούν να χρησιμοποιούνται πολλές φορές (reusability) σε διαφορετικές εφαρμογές, καθώς και οτι μπορούν εύκολα να δημιουργηθούν νέες κλάσεις που "κληρονομούν" τα δεδομένα και της μεθόδους μίας άλλης κλάσης (η ιδιότητα αυτή λέγεται κληρονομικότητα ή inheritance).

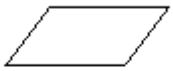
Η τεχνική αυτή συνδυάζει πλεονεκτήματα του δομημένου προγραμματισμού, με την επαναχρησιμοποίηση και ευκολία συντήρησης του κώδικα, παρουσιάζει ωστόσο αυξημένη δυσκολία και πολυπλοκότητα.

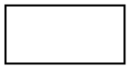
### 7.2 Αλγόριθμοι και λογικά διαγράμματα

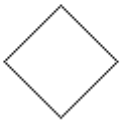
Μιλήσαμε ήδη για αλγόριθμους που χρησιμοποιούνται για την επίλυση ενός προβλήματος. Αλγόριθμος είναι μία σειρά από οδηγίες που δίνονται σε μορφή διακεκριμένων βημάτων για την επίλυση ενός προβλήματος. Ο αλγόριθμος είναι ανεξάρτητος από τη γλώσσα που θα χρησιμοποιήσουμε στη συνέχεια.

Για τη γραφική απεικόνιση ενός αλγόριθμου χρησιμοποιούμε το λογικό διάγραμμα ή διάγραμμα ροής (flow chart). Κάθε σύμβολο του λογικού διαγράμματος παριστάνει μία συγκεκριμένη λειτουργία. Υπάρχουν πάρα πολλά τέτοια σύμβολα αλλά τα σπουδαιότερα είναι τα παρακάτω (δίπλα δίνονται οι επεξηγήσεις τους) :

Αρχή ή τέλος διαδικασίας 

Γενικό Σύμβολο για είσοδο/έξοδο (input/output) δεδομένων ή αποτελεσμάτων αντίστοιχα 

Εκτέλεση κάποιας πράξης ή υπολογισμού 

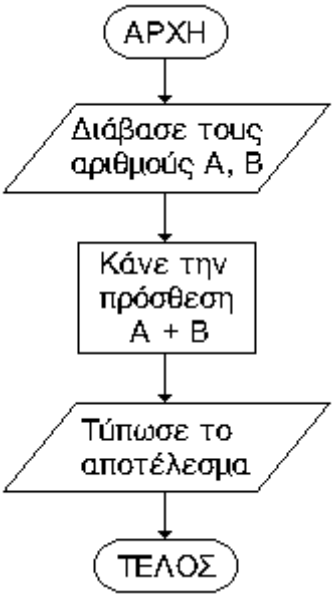
Διακλάδωση με έλεγχο κάποιας συνθήκης 

Σημείο σύνδεσης διαφορετικών κομματιών ενός λογικού διαγράμματος (συνήθως χρησιμοποιείται όταν δεν έχουμε αρκετό χώρο) 

Παρακάτω θα δούμε ορισμένα παραδείγματα όπου γίνεται χρήση λογικών διαγραμμάτων.

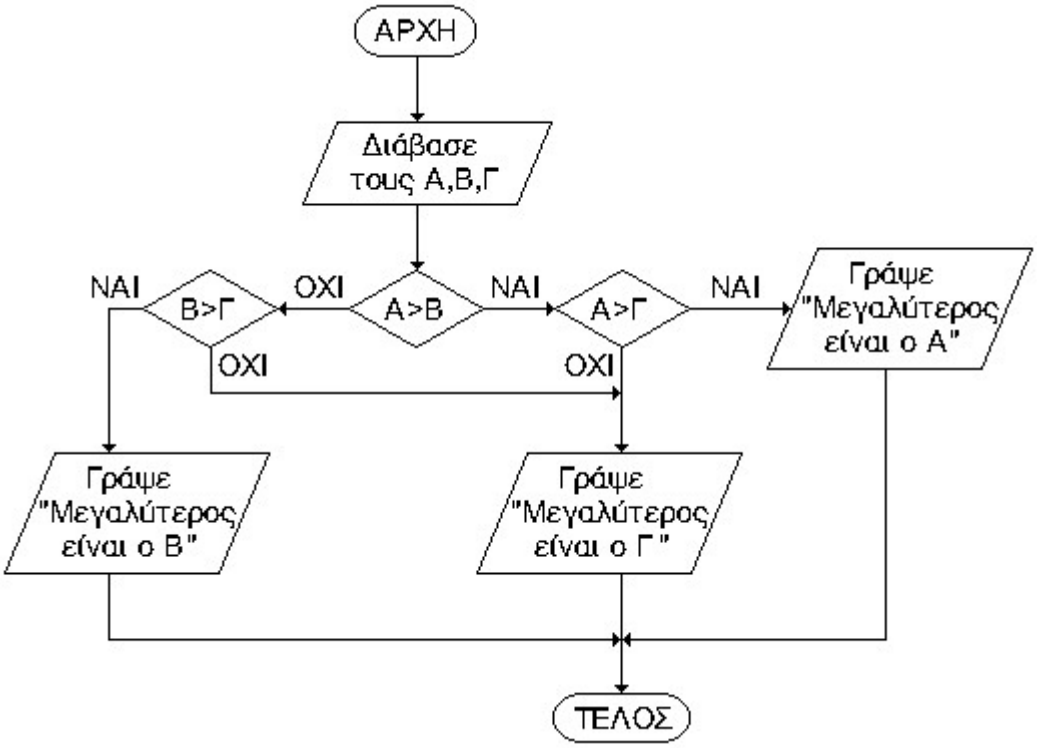
Παράδειγμα 1

Ζητείται το διάγραμμα ροής για την πρόσθεση δύο αριθμών Α,Β που δίνονται ως δεδομένα.



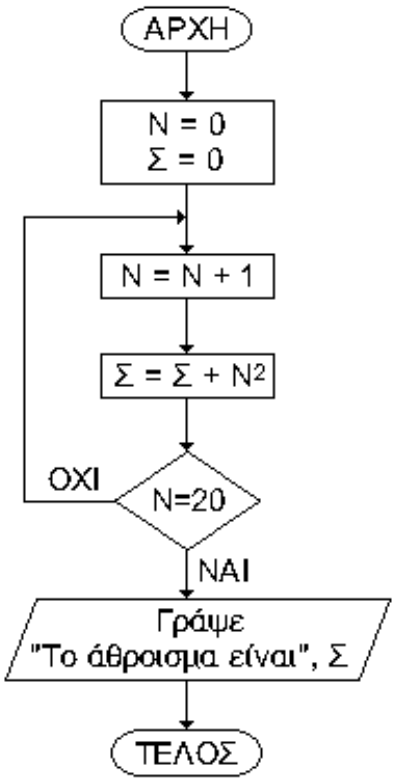

Παράδειγμα 2

Ζητείται το διάγραμμα ροής για να βρεθεί ο μεγαλύτερος μεταξύ τριών αριθμών Α,Β,Γ που δίνονται ως δεδομένα.



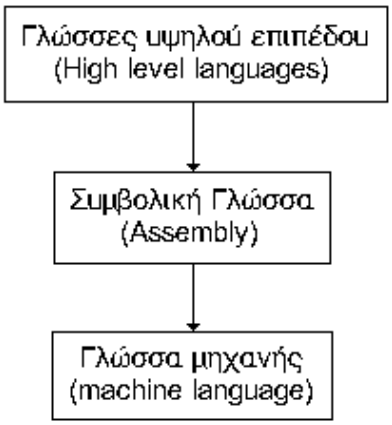

Παράδειγμα 3

Ζητείται το διάγραμμα ροής για τον υπολογισμό του αθροίσματος των τετραγώνων των 20 πρώτων φυσικών αριθμών.



### 7.3 Γλώσσες προγραμματισμού

Είδαμε ήδη οτι ο υπολογιστής καταλαβαίνει μόνο τη γλώσσα μηχανής (machine language) ενώ ο χρήστης γράφει (συνήθως) τα προγράμματα του σε κάποια γλώσσα υψηλού επιπέδου. Το πρόγραμμα αυτό ονομάζεται πηγαίο ή αρχικό πρόγραμμα (source program), το οποίο μεταφράζεται από το μεταφραστή (compiler) στο τελικό πρόγραμμα (object program) στη γλώσσα μηχανής. Για να γίνει αυτό όμως θα πρέπει να μην έχει συντακτικά λάθη, τα οποία αναγνωρίζει ο μεταφραστής. Τα λογικά λάθη όμως δεν είναι δυνατόν να ανιχνευτούν και διαπιστώνονται μόνο από το χρήστη από το γεγονός ότι θα οδηγήσει σε λάθος αποτελέσματα μετά την εκτέλεση.
Η ιεραρχία των γλωσσών προγραμματισμού είναι αυτή που φαίνεται στο παρακάτω σχήμα :




Οι γλώσσες υψηλού επιπέδου είναι βέβαια πολύ πιο εύκολες στην εκμάθηση και ο προγραμματισμός του υπολογιστή γίνεται ασύγκριτα πιο εύκολα για τον χρήστη από ότι με την assembly πολύ περισσότερο από ότι με τη γλώσσα μηχανής, Η πρώτη γλώσσα Υψηλού επιπέδου που παρουσιάστηκε ήταν η FORTRAN το 1957. Από τότε έχουν παρουσιαστεί πολλές γλώσσες κατάλληλες για την επίλυση γενικών και ειδικότερων προβλημάτων. Η ανάπτυξη των γλωσσών προγραμματισμού είναι ένας από τους πιο βασικούς κλάδους της επιστήμης των υπολογιστών και συνδέεται με την εξέλιξη της τεχνολογίας λογισμικού (software engineering). Μετά τις τις πρώτες γλώσσες που παρουσιάστηκαν, έγινε προσπάθεια και αναπτύχθηκαν γλώσσες που ενσωματώνουν διάφορες τεχνικές, όπως τον δομημένο προγραμματισμό ή τον τμηματικό προγραμματισμό.
Έχουμε ήδη αναφέρει ότι ο μεταφραστής (compiler) είναι ένα πρόγραμμα που δέχεται στην είσοδο του ένα ολόκληρο πρόγραμμα γραμμένο σε κάποια γλώσσα (source program) και παράγει στην έξοδο ένα αντίστοιχο τελικό πρόγραμμα (object program) σε γλώσσα μηχανής. Στη συνέχεια εκτελείται το τελικό πρόγραμμα. Ο μεταφραστής χρειάζεται μόνο στην αρχή για να κάνει τη μετάφραση του αρχικού προγράμματος.

Ο διερμηνέας (interpreter) είναι ένα πρόγραμμα της ίδιας κατηγορίας το οποίο δεν μεταφράζει ολόκληρο το αρχικό πρόγραμμα παράγοντας κάποιο τελικό πρόγραμμα σε γλώσσα μηχανής, αλλά παίρνει τις εντολές του μία προς μία και παράγει εκείνη τη στιγμή μία αντίστοιχη σειρά εντολών γλώσσας μηχανής που εκτελούνται. Η χρήση μεταφραστή δίνει στο πρόγραμμα μεγαλύτερη ταχύτητα εκτέλεσης, ενώ η χρήση διερμηνέα δίνει τη δυνατότητα της άμεσης διόρθωσης σφαλμάτων και γενικά δίνει αμεσότητα κατά την εκτέλεση.

Γλώσσες όπως η FORTRAN, η PASCAL, η C, η C++, η ADA χρησιμοποιούν μεταφραστή ενώ γλώσσες όπως η BASIC, η PROLOG, η PYTHON, η MATLAB χρησιμοποιούν διερμηνέα. Κάποιες γλώσσες όπως η JAVA χρησιμοποιούν υβριδικές τεχνικές.

### 7.3.1 Κατηγορίες γλωσσών προγραμματισμού

Όπως είπαμε έχουν αναπτυχθεί πολλές γλώσσες προγραμματισμού οι οποίες ανάλογα με τη χρήση τους και με την περιοχή όπου εφαρμόζονται χωρίζονται σε κατηγορίες. Ωστόσο ένας τέτοιος διαχωρισμός δεν είναι απόλυτος, και μία γλώσσα μπορεί να ανήκει σε περισσότερες από μία κατηγορίες.

Α) Γλώσσες γενικού σκοπού

Με μία τέτοια γλώσσα μπορούμε θεωρητικά να επιλύσουμε οποιοδήποτε πρόβλημα που μπορεί να λυθεί με υπολογιστή, ωστόσο ανάλογα με το είδος του προβλήματος μπορούμε να χωρίσουμε τις γλώσσες της κατηγορίας αυτής σε υποκατηγορίες :

α) Γλώσσες για επίλυση προβλημάτων και επιστημονικές εφαρμογές. Οι γλώσσες αυτές είναι κατάλληλες για επίλυση προβλημάτων με πολλούς υπολογισμούς ή που περιλαμβάνουν μαθηματικές συναρτήσεις κ.λ.π Διαθέτουν μεγάλο πλήθος τύπων αριθμητικών δεδομένων και δουλεύουν με πολλές αριθμητικές πράξεις και συναρτήσεις. ‘Εχουν όμως πολύ περιορισμένες δυνατότητες χειρισμού αρχείων δεδομένων (εισόδου-εξόδου και επεξεργασίας). Στην κατηγορία αυτή ανήκουν οι Visual Basic, C, C++, JAVA, PYTHON, MATLAB καθώς και οι παλαιότερες FORTRAN και PASCAL.

β) Γλώσσες για εμπορικές εφαρμογές. Οι γλώσσες αυτές χρησιμοποιούνται για την ανάπτυξη εμπορικών και εφαρμογών (όπως έσοδα-έξοδα, αποθήκη, πελάτες, μισθοδοσία κ.λ.π.) ή γενικώτερα εφαρμογές που χειρίζονται μεγάλο όγκο δεδομένων (όπως αρχεία υπαλλήλων, ασθενών, εξεταζόμενων κ.λ.π.). Έχουν αυξημένες δυνατότητες για είσοδο, επεξεργασία, έξοδο αρχείων αλλά περιορισμένες δυνατότητες για υπολογισμούς και μαθηματικές πράξεις. Γλώσσας αυτής της κατηγορίας είναι οι COBOL, RPG και άλλες. Ωστόσο τις τελευταίες δεκαετίες για την ανάπτυξη εφαρμογών αυτού του είδους χρησιμοποιούνται κυρίως ολοκληρωμένα πακέτα λογισμικού, που συνήθως αναφέρονται ως ERP (Enterprise Resource Planning).

γ) Υπάρχει μία άλλη κατηγορία γλωσσών προγραμματισμού, σημαντικότερες από τις οποίες είναι οι C και C++, οι οποίες εκτός από τις συνήθεις εντολές που χρησιμοποιούνται για

ανάπτυξη εφαρμογών, περιλαμβάνουν και εντολές που αναφέρονται στο Λειτουργικό Σύστημα ή στο υλικό (Hardware) του υπολογιστή. Οι γλώσσες αυτές συνδυάζουν χαρακτηριστικά γλώσσας υψηλού επιπέδου και συμβολικής γλώσσας.

Β) Γλώσσες ειδικού σκοπού

Μεγάλο πλήθος γλωσσών προγραμματισμού έχουν αναπτυχθεί για εξυπηρέτηση ειδικών εφαρμογών. Τέτοιες εφαρμογές είναι

α) Τεχνητή Νοημοσύνη (AI - artificial Intelligence). Οι γλώσσες αυτού του είδους μπορούν να χειρίζονται εύκολα δεδομένα που δίνονται σε κάποια συμβολική μορφή. Εφαρμόζουν τις τεχνικές της συμπερασματολογίας και της αναζήτησης μέσω κανόνων-γεγονότων. Προβλήματα που εμπίπτουν στο πεδίο της τεχνητής νοημοσύνης είναι η απόδειξη θεωρημάτων, το παίξιμο παιχνιδιών, κατανόηση φυσικής γλώσσας, τεχνητή ομιλία κ.α. Γλώσσες που χρησιμοποιούνται για εφαρμογές αυτής της κατηγορίας είναι οι PYTHON, JAVA, C++, καθώς και οι παλαιότερες LISP και PROLOG.

β) Έλεγχος μηχανών – Ρομποτική. Υπάρχουν γλώσσες ειδικά σχεδιασμένες για τον προγραμματισμό μηχανών και ρομπότ που συνήθως χρησιμοποιούνται στη βιομηχανική παραγωγή. Στις περιπτώσεις αυτές έχουμε επεξεργασία σε πραγματικό χρόνο (real time). Υπάρχουν εξειδικευμένες γλώσσες για εφαρμογές αυτής της κατηγορίες αλλά και οι γλώσσες C, C++, JAVA, PYTHON, MATLAB χρησιμοποιούνται επίσης.

γ) Εφαρμογές γραφικών (graphics) Με τις γλώσσες αυτές μπορούμε να δημιουργούμε ευκολότερα γραφικές απεικονίσεις, δισδιάστατες και τρισδιάστατες.

δ) Συστήματα Διοίκησης Βάσεων Δεδομένων (ΣΔΒΔ ή DBMS : Data Base Management Systems). Στις εφαρμογές αυτές αυτού του είδους έχουμε μεγάλο πλήθος δεδομένων που έχουν κάποια οργάνωση και λογικές διασυνδέσεις μεταξύ τους. Σε ένα ΣΔΒΔ υπάρχουν ειδικές γλώσσες με τις οποίες ο χρήστης περιγράφει τη μορφή που παίρνουν τα δεδομένα, χειρίζεται τα δεδομένα ή ζητάει την έξοδο κάποιων δεδομένων υποβάλλοντας ερωτήσεις (queries) πάνω στο το περιεχόμενο της Βάσης Δεδομένων, με τη χρήση ειδικών γλωσσών (query languages).

ε) Γλώσσες οπτικού προγραμματισμού (visual programming languages). Είναι ιδιαίτερα φιλικές στο χρήστη, καθώς ο προγραμματισμός γίνεται μετακινώντας αντικείμενα στην οθόνη.

στ) Γλώσσες που χρησιμοποιούνται για κατασκευή ιστιοσελίδων (websites), όπως η HTML (Hypertext Markup Language) και η νεώτερη HTML5.

Εκτός από τις παρακάτω παραπάνω υπάρχουν βέβαια και άλλες πολυάριθμες εφαρμογές όπου χρησιμοποιούνται αντίστοιχες γλώσσες προγραμματισμού. Γλώσσες όπως οι PERL, RUBY και παλαιότερα η SNOBOL χρησιμοποιούνται για χειρισμό κειμένων ή συμβολοσειρών. Υπάρχει μία κατηγορία γλωσσών που λέγονται μεταφραστές μεταφραστών (compiler compilers) που παίρνουν σαν είσοδο την περιγραφή μιας νέας γλώσσας προγραμματισμού που θέλουμε να δημιουργήσουμε και δίνουν στην έξοδο ένα πρόγραμμα

που αποτελεί μερική υλοποίηση του μεταφραστή της γλώσσας. Άλλες γλώσσες χρησιμοποιούνται για εκπαιδευτικούς σκοπούς, για σχεδίαση ολοκληρωμένων κυκλωμάτων κ.α.

# ΚΕΦΑΛΑΙΟ 8. ΔΕΔΟΜΕΝΑ ΚΑΙ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

## 8.1 Η έννοια των δεδομένων (data) και η παράστασή τους στον Η/Υ

Όταν γράφουμε ένα πρόγραμμα, μεταφέρουμε στον υπολογιστή τη λύση ενός προβλήματος. Το πρόβλημα αυτό γενικά θα περιλαμβάνει αντικείμενα και λειτουργίες δηλαδή αλλαγές και αλληλεπιδράσεις μεταξύ των αντικειμένων. Μία γλώσσα προγραμματισμού διευκολύνει αυτήν την διεργασία, παρέχοντας τα κατάλληλα εργαλεία για την απεικόνιση των αντικειμένων και των λειτουργιών. Για την απεικόνιση των αντικειμένων χρησιμοποιούνται τα δεδομένα (ή δομές δεδομένων) . Σύμφωνα με έναν άλλον ορισμό δεδομένο είναι ότι μπορεί να επεξεργαστεί από τον υπολογιστή. Στο επίπεδο του υλικού (hardware) τα δεδομένα καταχωρούνται στην ΚΜ (Κεντρική Μνήμη) ή στις περιφερειακές μονάδες μνήμης του υπολογιστή από μορφή δυαδικών ψηφίων (bits) και αναπαριστώνται σύμφωνα με κάποιον κώδικα και αναπαριστώνται σύμφωνα με κάποιον κώδικα (π.χ. ASCII), όπως είδαμε στο κεφάλαιο 3.

Στις διάφορες γλώσσες προγραμματισμού ως φορείς απλών δεδομένων χρησιμοποιούνται οι σταθερές (constants) και οι μεταβλητές (variables). Η διαφορά είναι ότι η τιμή της μεταβλητής μεταβάλλεται κατά την εκτέλεση του προγράμματος. Τύπος δεδομένων (data type) είναι ένα σύνολο τιμών που μπορεί να πάρει μία μεταβλητή. Κάθε μεταβλητή (ή σταθερά) αντιστοιχεί σε ένα μόνο τύπο, ο οποίος προσδιορίζει τις τιμές που μπορεί να πάρει, καθώς και τις πράξεις που μπορούν να πραγματοποιηθούν πάνω σε αυτήν. Συνήθως η γλώσσα προγραμματισμού απαιτεί από το χρήστη να δηλώσει εκ των προτέρων τον τύπο κάθε μεταβλητής έστω και κατά συνθήκη (π.χ. από τον το πρώτο γράμμα του ονόματός της μεταβλητής).

## 8.2 Μελέτη των δεδομένων. Τύποι δεδομένων

### 8.2.1 Απλοί ή βαθμωτοί τύποι

Όλες οι γλώσσες προγραμματισμού επιτρέπουν στο χρήστη να ορίζει, να δημιουργεί και να χειρίζεται μεταβλητές που ανήκουν σε κάποιον από τους παρακάτω βασικούς τύπους :

α) Ακέραιοι (integer)

Οι επιτρεπτές τιμές αυτού του τύπου είναι οι θετικοί και αρνητικοί ακέραιοι αριθμοί που η απόλυτη τιμή τους δεν υπερβαίνει κάποιο όριο. Συνήθως παριστάνεται με μία λέξη του υπολογιστή.

β) Πραγματικοί (real)

Οι η επιτρεπτές τιμές αυτού του τύπου είναι θετικοί και αρνητικοί ρητοί με πεπερασμένο αριθμό δεκαδικών ψηφίων. Γράφονται είτε με υποδιαστολή είτε σε εκθετική μορφή , ενώ η

παράσταση τους στον υπολογιστή γίνεται με κάποια μέθοδο κινητής υποδιαστολής( floating-point). Πολλές γλώσσες επιτρέπουν και πραγματικούς αριθμούς διπλής προσέγγισης (double precision) για ακριβείς αριθμητικούς υπολογισμούς.

γ) Λογικοί ή δυαδικοί (boolean)

παίρνουν δύο μόνο τιμές TRUE ή FALSE (αντιστοιχούν στις τιμές 'αληθής' και 'ψευδής' της άλγεβρας Boole που είδαμε στο Κεφάλαιο 3. Είναι φανερό ότι ένα μόνο bit αρκεί για να παραστήσει την τιμή μιας τέτοιας μεταβλητής αλλά πολλές φορές είναι πιο εύκολο να τους διατεθεί ένα ολόκληρο byte ή ακόμα και μία λέξη. Στους προηγούμενους τύπους που συναντήσαμε εφαρμόζονται αλγεβρικές πράξεις και συναρτήσεις, στις μεταβλητές λογικού τύπου εφαρμόζονται οι λογικές πράξεις της άλγεβρας Boole (NOT, AND, OR)

δ) Χαρακτήρες (characters)

Οι τιμές αυτού του τύπου είναι όλοι οι δυνατοί χαρακτήρες σύμφωνα με κάποιο σύστημα κωδικοποίησης (συνήθως του κώδικα ASCII). Παριστάνονται με ένα byte του υπολογιστή.

ε) Συμβολοσειρες (strings)

Οι τιμές αυτού του τύπου είναι ακολουθίες χαρακτήρων, συνήθως με κάποιο άνω όριο μήκους ή και χωρίς άνω όριο μήκους. Παριστάνονται σαν διαδοχικά bytes στον υπολογιστή

στ) Άτομα (atoms)

Σε ορισμένες γλώσσες προγραμματισμού χρησιμοποιείται αυτός ο τύπος δεδομένων, που είναι μια βασική μονάδα δεδομένων που δεν διαιρείται, πχ μια σταθερά ή μια συμβολοσειρά. Η τιμή ενός τέτοιου τύπου μεταβλητής της είναι το ίδιο της το όνομα (δηλαδή όπως λέμε αυτοαποτιμούνται). Συνήθως υλοποιούνται στον υπολογιστή σαν συμβολοσειρές.

ζ) Δείκτες (pointers)

Οι επιτρεπτές τιμές αυτού του τύπου είναι μία διεύθυνση μέσα στο χώρο εργασίας του προγράμματος. Υλοποιούνται στον υπολογιστή σαν λέξεις.

Ο μεταφραστής (compiler) της γλώσσας προγραμματισμού φροντίζει για τον πιο αποδοτικό τρόπο καταχώρησης των μεταβλητών στον υπολογιστή ανάλογα με τον τύπο τους

### 8.2.2 Σύνθετοι τύποι δεδομένων ή Δομές Δεδομένων

Οι σύνθετοι τύποι ή δομές δεδομένων συντίθεται κατά του έναν ή τον άλλον τρόπο από απλούστερους. Όταν τα απλούστερα δεδομένα που συνθέτουν τη δομή είναι του ιδίου τύπου, λέμε ότι η δομή είναι ομοιογενής, αλλιώς ότι είναι ετερογενής. Επίσης ο αριθμός απλών δεδομένων μιας δομής είναι σταθερός η μεταβλητός.

Ο πιο συνηθισμένος ίσως σύνθετος, ομογενής και σταθερός τύπος είναι ο πίνακας (array), τον οποίον θα εξετάσουμε αναλυτικά πιο κάτω. Ένας άλλος χρήσιμος σύνθετος τύπος δεδομένων είναι η εγγραφή η εγγράφημα (record). Η εγγραφή είναι δομή σταθερού πλήθους (μήκους) αλλά ετερογενής, διότι τα μέρη που την αποτελούν και που ονομάζονται πεδία

(fields), μπορεί να είναι διαφορετικού τύπου, όπως π.χ. η εγγραφή που περιλαμβάνει κάποια στοιχεία ενός υπαλλήλου, όπως ονοματεπώνυμο (θα είναι πίνακας χαρακτήρων), ηλικία θα είναι ακέραιος (1….100), άν έχει εκπληρώσει τη στρατιωτική του θητεία (θα είναι λογική μεταβλητή), μισθός (θα είναι ακέραια ή πραγματική μεταβλητή).
Ένα σύνολο εγγραφών αποτελούν ένα αρχείο (file) που είναι ομογενής τύπος δεδομένων μεταβλητού μήκους.

### 8.2.3 Μελέτη της μορφής μιας δομής δεδομένων

Μία οποιαδήποτε δομή δεδομένων θα αποτελείται από ένα σύνολο κόμβων (nodes). Κάθε κόμβος περιλαμβάνει δεδομένα που αντιστοιχούν σε μία ή περισσότερες μεταβλητές, που αντίστοιχα καταλαμβάνουν μία ή περισσότερες λέξεις του Η/Υ. Κάθε μία από τις μεταβλητές του κόμβου λέγεται πεδίο (field) και μπορεί να είναι είτε δεδομένα (data) είτε κάποιος δείκτης (pointer) ο οποίος δείχνει τη διεύθυνση ενός κόμβου. Ο κόμβος μπορεί να περιλαμβάνει κανένα, ένα ή περισσότερα πεδία (δεδομένα ή δείκτες). Έτσι ο πίνακας είναι μια δομή δεδομένων που οι κόμβοι του περιέχουν μόνο δεδομένα (όχι δείκτες), ενώ στη γραμμική λίστα η διαδοχικότητα των κόμβων απεικονίζεται είτε με την τοποθέτηση σε συνεχόμενες θέσεις μνήμης είτε με χρήση δείκτη σε κάθε κόμβο.

### 8.2.4 Βασικές λειτουργίες σε μία δομή δεδομένων.

Οι βασικές λειτουργίες ή πράξεις που μας ενδιαφέρει συνήθως να εφαρμόσουμε στα στοιχεία (δηλαδή τους κόμβους) μιας δομής δεδομένων είναι οι ακόλουθες:
1) Προσπέλαση (access) Είναι η αναζήτηση ενός κόμβου είτε για να εξετάσουμε τα περιεχόμενα του για να ενημερώσουμε (update) τα περιεχόμενα του
2) Εισαγωγή (insert) ενός νέου κόμβου σε μία υπάρχουσα δομή δεδομένων
3) Διαγραφή (delete) ενός υπάρχοντος κόμβου.
4) Αναζήτηση (search) δηλαδή η προσπέλαση ενός ή περισσοτέρων κόμβων για τους οποίους κάποιου ή κάποια από τα πεδία έχουν κάποια συγκεκριμένη ή κάποιες συγκεκριμένες τιμές. Το πεδίο βάσει του οποίου γίνεται η αναζήτηση λέγεται το κλειδί (key)
5) Ταξινόμηση (sorting) των κόμβων κατά την αύξουσα ή τη φθίνουσα σειρά κάποιου κλειδιού
6) Αντιγραφή (copy) όλων ή κάποιων από τους κόμβους μίας δομής σε μία άλλη
7) Συνένωση (merge) δύο ή περισσότερων δομών σε μία ενιαία
Πολλές φορές μπορεί μία δομή δεδομένων να είναι αποδοτική σε κάποια από τις ανωτέρω λειτουργίες ενώ για κάποια άλλη λειτουργία να είναι πιο αποδοτική κάποια άλλη δομή δεδομένων.

## 8.3 Δομές δεδομένων και αλγόριθμοι

Σε ένα σύστημα επεξεργασίας δεδομένων η μορφή της δομής δεδομένων που χρησιμοποιείται βρίσκεται σε αλληλοεξάρτηση με τον αλγόριθμο που χρησιμοποιείται, ώστε μπορούμε να πούμε ότι ένα πρόγραμμα περιλαμβάνει έναν αλγόριθμο και μία δομή δεδομένων (αλγόριθμοι + δομές δεδομένων = προγράμματα, σύμφωνα με τη διατύπωση του N.Wirth)

Για την επίλυση προβλημάτων που σχετίζονται με τις δομές δεδομένων (π.χ. αναζήτηση ή ταξινόμηση) χρησιμοποιούνται διάφοροι αλγόριθμοι. Μας ενδιαφέρει γενικά να μπορέσουμε να εκτιμήσουμε την αποδοτικότητα (efficiency) των αλγορίθμων. Μία πρώτη προσέγγιση για να μετρήσουμε την αποδοτικότητα ενός αλγορίθμου, έχουμε με το να μετρήσουμε είτε το χρόνο επεξεργασίας είτε τη χωρητικότητα μνήμης που απαιτεί ένα πρόγραμμα που υλοποιεί τον αλγόριθμο, το οποίο εκτελείται με κάποια δεδομένα. Η μέθοδος αυτή έχει το μειονέκτημα ότι το αποτέλεσμα που θα πάρουμε εξαρτάται από το συγκεκριμένο πρόγραμμα και τα συγκεκριμένα δεδομένα και όχι μόνο από την αποδοτικότητα του ίδιου του αλγόριθμου.

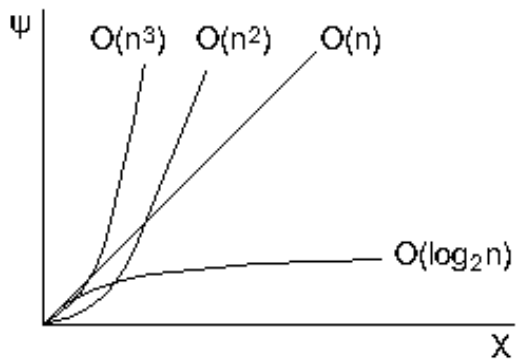
Μία άλλη προσέγγιση στο πρόβλημα είναι να ορίσουμε για κάποιο συγκεκριμένο πρόβλημα μία μεταβλητή που εκφράζει το μέγεθος (size) του προβλήματος. Η μεταβλητή αυτή θα εξαρτάται από το συγκεκριμένο πρόβλημα (π.χ. για την ταξινόμηση k στοιχείων θα είναι n = k). Στη συνέχεια προσπαθούμε να εκφράσουμε το χρόνο επεξεργασίας t και την απαιτούμενη χωρητικότητα μνήμης s σαν συνάρτηση του n, π.χ. t=f(n), s=g(n). Στις περισσότερες περιπτώσεις δεν μας ενδιαφέρει η ακριβής έκφραση των συναρτήσεων f(n) και g(n) του αλλά η γενική συμπεριφορά των t και s σαν συνάρτηση του n. Έτσι έχουμε τον O-συμβολισμό

$t \cong O(f(n))$ $s \cong O(g(n))$

Οι συναρτήσεις f(n) και g(n) εκφράζουν τη χρονική πολυπλοκότητα και την πολυπλοκότητα χώρου αντίστοιχα του συγκεκριμένου αλγόριθμου. Παραδείγματα εκφράσεων της πολυπλοκότητας ενός αλγόριθμου σαν συνάρτηση του μεγέθους (n) είναι οι μορφές

$O(\log_2 n)$, $O(n)$, $O(n^2)$, $O(n^3)$ κ.λ.π.

Στο παρακάτω σχήμα φαίνεται και η γραφική παράσταση των παραπάνω εκφράσεων (άξονας των χ : μέγεθος προβλήματος, άξονας των ψ : απαιτούμενος χώρος μνήμης ή χρόνος επεξεργασίας).




Οι αλγόριθμοι των οποίων η πολυπλοκότητα εκφράζεται σαν δύναμη του n, π.χ. O(n), $O(n^{3/2})$, $O(n^2)$, λέγονται πολυωνυμικοί, ενώ αλγόριθμοι των οποίων η πολυπλοκότητα έχει εκφράσεις της μορφής ($O(2^n)$, $O(n^2*2^n)$ ή $O(n^n)$ λέγονται μη πολυωνυμικοί.

## 8.4 Πίνακες

### 8.4.1 Γενικά

Ο πίνακας είναι μία απλής μορφής δομή δεδομένων που περιλαμβάνει ένα σύνολο στοιχείων του ιδίου τύπου. Είναι δομή δεδομένων ομοιογενής, σταθερού μήκους. Η δήλωση του πίνακα και η αναφορά στα στοιχεία του εξαρτάται από τη γλώσσα προγραμματισμού. 'Ετσι στην C++ η δήλωση int mytable [4] δηλώνει έναν πίνακα με τέσσερις ακέραιους αριθμούς. Η αναφορά mytable [2] σημαίνει το 3ο στοιχείο του Πίνακα, δεδομένου οτι η αρίθμηση γραμμών ενός πίνακα ν στοιχείων στη C++ αρχίζει από το 0 (και φτάνει ως ν-1). Ο αριθμός που δείχνει την τάξη του στοιχείου μέσα στον πίνακα λέγεται δείκτης (index).

Μπορούμε να έχουμε πίνακα δύο διαστάσεων. Η δήλωση ενός πίνακα 4 γραμμών και 5 στηλών ακεραίων αριθμών.στην C++ είναι : int mytable[4] [5] και η αναφορά mytable[1] [3] σημαίνει το στοιχείο (ακέραιος αριθμός) που βρίσκεται στη διασταύρωση της 2ης γραμμής και της 4ης στήλης του δισδιάστατου αυτού πίνακα,

Ομοίως ορίζονται οι πολυδιάστατοι πίνακες.

Το πλήθος των στοιχείων ενός πίνακα πρέπει να καθορίζεται από το εκάστοτε πρόγραμμα. ΄Ενα άλλο χαρακτηριστικό των πινάκων είναι ότι τα στοιχεία τους καταχωρούνται συνήθως σε συνεχόμενες θέσεις μνήμης, πράγμα το οποίο θα υποθέσουμε στη συνέχεια.

Στη συνέχεια θα εξετάσουμε πώς καταχωρούνται τα στοιχεία ενός πίνακα σε διαδοχικές θέσεις μνήμης. Στις περισσότερες γλώσσες προγραμματισμού ο απαραίτητος χώρος μνήμης για τον πίνακα καθορίζεται κατά τη μετάφρασή (compilation).

Έστω οτι έχουμε τον μονοδιάστατο πίνακα A[u] (u : upper bound = άνω όριο). Άν κάθε στοιχείο του πίνακα καταλαμβάνει μία θέση μνήμης, για την καταχώρηση του πίνακα απαιτούνται u διαδοχικές θέσεις μνήμης. Αν υποτεθεί ότι η αρχική θέση που καταλαμβάνει ο πίνακας στη μνήμη του Η/Υ έχει διεύθυνση Δ(Ι) τότε η διεύθυνση του i-στού στοιχείου του Πίνακα στη μνήμη του θα δίνεται από τη σχέση :

$$\Delta(i) = \Delta(I) + i - 1 \qquad (1)$$

Αν κάθε στοιχείο του πίνακα καταλαμβάνει a θέσεις μνήμης, τότε η παραπάνω σχέση (1) γίνεται :

$$\Delta(i) = \Delta(I) + a*(i - 1) \qquad (2)$$

Έστω ότι έχουμε δισδιάστατο πίνακα, δηλαδή τον Πίνακα A [u1] [u2]. Υποθέτουμε επίσης ότι ο πίνακας καταχωρείται στη μνήμη κατά γραμμές, δηλαδή καταχωρείται πρώτα η πρώτη γραμμή του πίνακα μετά τη δεύτερη κ.ο.κ ως την τελευταία. (άλλος τρόπος είναι η καταχώριση κατά στήλες). Αν Δ είναι η διεύθυνση της αρχικής θέσης που καταλαμβάνει ο πίνακας και κάθε στοιχείο του καταλαμβάνει a θέσεις μνήμης, τότε η διεύθυνση του στοιχείου A[i] [j] του πίνακα θα προσδιορίζεται από τη σχέση :

$$\Delta(i,j) = \Delta + a*(i - 1)*u2 + a*(j - 1) \qquad (3)$$

Οι παραπάνω σχέσεις (1), (2) και (3) λέγονται συναρτήσεις απεικόνισης, και χρησιμοποιούνται από τον τον μεταφραστή για τον προσδιορισμό της θέσης μέσα στη μνήμη ενός στοιχείου του πίνακα, κάθε φορά που το πρόγραμμα αναφέρεται σε κάποιο στοιχείο του πίνακα. Μπορούν φυσικά να γενικευτούν και για v-διάστατους πίνακες .

### 8.4.2 Ειδικές μορφές πινάκων

Για έναν πίνακα n x n απαιτούνται στή γενική περίπτωση $n^2$ θέσεις μνήμης για την καταχώρισή του (υποθέτουμε ότι κάθε στοιχείο του πίνακα καταλαμβάνει μία θέση μνήμης). Όταν όμως ο πίνακας έχει κάποια ειδική μορφή μπορούμε να εκμεταλλευτούμε το γεγονός αυτό ώστε να ελαττώσουμε χώρο φύλαξης του πίνακα αυτού. Εξετάζουμε μερικές ειδικές μορφές πινάκων

1) Συμμετρικοί και τριγωνικοί πίνακες.

Ο συμμετρικός πίνακας έχει τη μορφή :

$$\begin{matrix} \alpha(1,1) & \alpha(2,1) & \ldots & \alpha(n,1) \\ \alpha(2,1) & \alpha(2,2) & \ldots & \alpha(n,2) \\ \ldots & \ldots & \ldots & \ldots \\ \alpha(n,1) & \alpha(n,2) & \ldots & \alpha(n,n) \end{matrix}$$

Δηλαδή ισχύει α(i,j) = a(j,i) Επειδή ο πίνακας αυτός έχει 1+2+ … +n = n*(n+1)/2 διαφορετικά στοιχεία, απαιτούνται για την καταχώρισή του n*(n+1)/2 θέσεις μνήμης.

Τα παραπάνω εφαρμόζονται και για τριγωνικούς πίνακες. Η μορφή ενός κάτω τριγωνικού πίνακα είναι :

α(1,1)
α(2,1) α(2,2)
.. .. ...
α(n,1) α(n,2) ... α(n,n)

2) Τριδιαγώνιοι πίνακες.

Ο τριδιαγώνιος πίνακας έχει τη μορφή :

| | | | | |
|---|---|---|---|---|
| α(1,1) | α(1,2) | 0 | ... | 0 |
| α(2,1) | α(2,2) | α(2,3) | ... | 0 |
| 0 | α(3,2) | α(3,3) | ... | 0 |
| … | … | … | ... | |
| 0 | 0 | 0 | … | α(n,n) |

δηλαδή όλα τα στοιχεία του είναι μηδενικά εκτός από τα στοιχεία της κύριας διαγωνίου και των άλλων δύο γειτονικών διαγωνίων.

Επειδή ο πίνακας αυτός έχει n +2*(n-1) = 3*n-2 μη μηδενικά στοιχεία, απαιτούνται για την καταχώρησή του 3*n-2 θέσεις μνήμης.

### 8.4.3 Πίνακες συμβολοσειρών

Ένας πίνακας με στοιχεία μεταβλητές τύπου χαρακτήρα είναι ουσιαστικά μία συμβολοσειρά (string).

Έστω οτι θέλουμε να καταχωρήσουμε ένα string με n χαρακτήρες. Κάθε χαρακτήρας χρειάζεται 1 byte για την καταχώρισή του. Μία θέση μνήμης όμως (μία λέξη) του Η/Υ περιλαμβάνει συνήθως περισσότερα bytes, άρα μπορεί να αποθηκεύσει περισσότερους, έστω s χαρακτήρες. Αν χρησιμοποιήσουμε μία θέση μνήμης ανά χαρακτήρα θα χρησιμοποιηθούν n θέσεις μνήμης για την καταχώρησή του string ενώ άν σε κάθε θέση μνήμης τοποθετηθούν δύο ή περισσότεροι χαρακτήρες, θα χρησιμοποιηθούν n/s θέσεις μνήμης, τεχνική η οποία ονομάζεται συμπύκνωση (packing). Στη γενική περίπτωση με την τεχνική αυτή θα χρειαστούν [n/s] θέσεις μνήμης, όπου [α] εκεί είναι η στρογγυλοποίηση του αριθμού α στον αμέσως επόμενο ακέραιό του.

Είναι φανερό ότι με το δεύτερο ορισμό επιτυγχάνουμε μείωση του απαιτούμενου χώρου μνήμης. Είναι όμως πιο δύσκολη η προσπέλαση του στοιχείου s[i], διότι θα πρέπει να

εντοπιστεί πρώτα η λέξη (θέση μνήμης) όπου αυτό βρίσκεται καταχωρημένο και εν συνεχεία τη θέση του μέσα στη λέξη. Σαν αποτέλεσμα επιβαρύνεται ο χρόνος επεξεργασίας.

## 8.5 Γραμμικές λίστες

### 8.5.1. Γενικά

Μία γραμμική λίστα (linear list) ορίζεται γενικά σαν ένα σύνολο κόμβων k(1), k(2), …., k(n), όπου υπάρχει ένας κόμβος k(1) που είναι πρώτος ενώ για κάθε κόμβο k(m) υπάρχει κάποιος προηγούμενος K(m-1) και κάποιος επόμενος k(m+1), όπου 1<m<n.
Μία λίστα παριστάνεται με το ακόλουθο σχήμα :



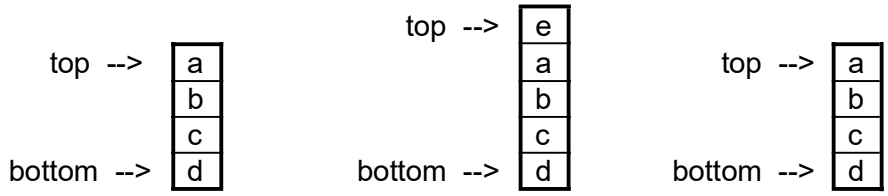

Κάθε κύτταρο (cell) της λίστας αποτελείται από δύο τμήματα. Το πρώτο τμήμα περιέχει κάποια πληροφορία και το δεύτερο περιέχει κάποιο δείκτη προς το επόμενο κύτταρο της λίστας. Ο δείκτης του τελευταίου κυττάρου της λίστας δεν δείχνει πουθενά (nil), ενώ ο δείκτης head δείχνει το πρώτο κύτταρο. Για να βρούμε ένα κύτταρο της λίστας με βάση τον δείκτη head πηγαίνουμε στο πρώτο κύτταρο και από εκεί ακολουθώντας τους δείκτες next φτάνουμε στο κύτταρο που θέλουμε.
Οι γραμμικές λίστες χωρίζονται σε σειριακές γραμμικές λίστες και σε συνδεδεμένες γραμμικές λίστες . Η βασική τους διαφορά είναι ότι οι σειριακές λίστες κατά την υλοποίησή τους στον Η/Υ καταχωρούνται σε συνεχόμενες θέσεις μνήμης, όπως οι πίνακες, ενώ κατά την υλοποίηση των συνδεδεμένων λιστών οι κόμβοι τους καταχωρούνται σε απομακρυσμένες θέσεις μνήμης, οι οποίες διασυνδέονται με δείκτες. Ακόμη, οι σειριακές λίστες είναι συνήθως στατικές, δηλαδή το μέγεθός τους καθορίζεται στο πρόγραμμα, ενώ οι συνδεδεμένες λίστες είναι συνήθως δυναμικές δηλαδή μπορούν να μικραίνουν ή να μεγαλώνουν κατά την εκτέλεση του προγράμματος, σύμφωνα με τις απαιτήσεις.

### 8.5.2 Σειριακές γραμμικές λίστες

Παρακάτω θα εξετάσουμε τις σημαντικότερες δομές που υλοποιούνται με σειριακές γραμμικές λίστες.

### 8.5.2.1 Στοίβα

Στη δομή της στοίβας (stack), που μπορούμε να την αντιπαραβάλουμε μία στοίβα από πιάτα θεωρούμε ότι υπάρχει ένας δείκτης στην κορυφή (top) της στοίβας και ένας δείκτης στον πυθμένα (bottom) της στοίβας. Η εισαγωγή ενός νέου στοιχείου γίνεται στην κορυφή της στοίβας, ενώ η εξαγωγή ενός στοιχείου γίνεται πάλι από την κορυφή της στοίβας και συνεχίζεται προς τα κάτω. Δηλαδή στη στοίβα εφαρμόζεται η τεχνική LIFO (Last In First Out).



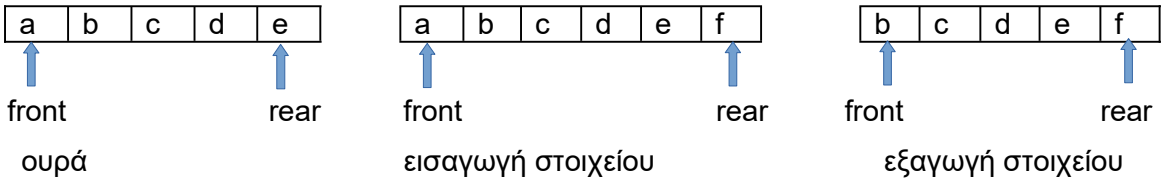

Στοίβα εισαγωγή στοιχείου εξαγωγή στοιχείου

Κατά την εισαγωγή (push) ενός στοιχείου στην κορυφή της στοίβας πρέπει να ελέγχεται αν η στοίβα είναι γεμάτη, ώστε να αποφευχθεί η υπερχείλιση (overflow). Αντίστοιχα κατά την εξαγωγή (pop) στοιχείου από τη στοίβα πρέπει να ελέγχεται αν υπάρχει τουλάχιστον ένα στοιχείο για εξαγωγή.

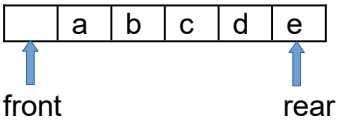
Ένας τρόπος υλοποίησης της στοίβας είναι με τη χρήση μονοδιάστατου πίνακα. Η στοίβα εφαρμόζεται σε πολλές περιπτώσεις στον προγραμματισμό. Χαρακτηριστικό παράδειγμα είναι η χρήση στοίβας κατά την κλήση υπορουτινών (subroutines ή procedures) και λειτουργιών (functions) σε μία γλώσσα προγραμματισμού. Όταν καλείται μία υπορουτινα, η αμέσως επόμενη διεύθυνση του κυρίως προγράμματος (διεύθυνση επιστροφής) καταχωρείται από τον μεταφραστή σε μία στοίβα. Όταν τελειώσει η εκτέλεση της υπορουτίνας, ο έλεγχος του προγράμματος μεταφέρεται στη διεύθυνση επιστροφής και αυτή απομακρύνεται από τη στοίβα. Όταν μία υπορουτίνα καλείται μέσα από κάποια άλλη, και μία τρίτη υποτοουτίνα καλείται μέσα από αυτήν κ.ο.κ. , είναι προφανές η χρησιμότητα της στοίβας και της τεχνικής LIFO. Ακόμη, η στοίβα είναι συνδεδεμένη με την έννοια της αναδρομής (recursion), δηλαδή όταν μία υπορουτίνα καλείται από τον εαυτό της (δηλαδή μέσα από το block των εντολών της).

### 8.5.2.2 Ουρά

Στη δομή της ουράς (queue) θεωρούμε ότι υπάρχουν δύο δείκτες στα δύο άκρα, ο εμπρός (front) δείκτης και ο πίσω (rear). Η εισαγωγή ενός νέου στοιχείου γίνεται στο πίσω άκρο, ενώ η εξαγωγή στοιχείου γίνεται από το εμπρός άκρο και συνεχίζεται προς τα πίσω. Δηλαδή στην ουρά εφαρμόζεται η τεχνική FIFO (First In First Out).




Όταν η ουρά περιέχει ένα μόνο στοιχείο ο εμπρός και ο πίσω δείκτης δείχνουν το ίδιο στοιχείο. Επειδή όμως τη συνθήκη αυτή τη χρησιμοποιούμε συνήθως για να δείξουμε ότι η ουρά είναι άδεια, υποθέτουμε ότι ο εμπρός δείκτης παίρνει τιμή που είναι κατά μία μονάδα μικρότερη από την αρχή της ουράς, δηλαδή :



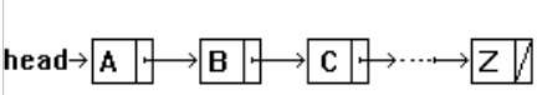

### 8.5.2.3 Ουρά με δύο άκρα

Η ουρά με δύο άκρα (double ended queue) είναι μία ουρά στην οποία επιτρέπεται να εισάγουμε και να διαγράφουμε στοιχεία και από τα δύο άκρα της ουράς. Μπορούμε επομένως να έχουμε εισαγωγή στοιχείου στο τέλος, εξαγωγή στοιχείου από την αρχή, εισαγωγή στοιχείου στην αρχή και εξαγωγή στοιχείου από το τέλος της ουράς.

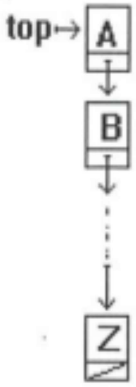
Όπως είπαμε ήδη, οι σειριακές γραμμές γραμμικές λίστες όπως και οι πίνακες καταχωρούνται σε διαδοχικές θέσεις μνήμης στον Η/Υ. Αυτό δημιουργεί ορισμένα προβλήματα όπως π.χ. στην εισαγωγή και εξαγωγή στοιχείων (εκτός αν πρόκειται για το πρώτο ή το τελευταίο στοιχείο). Οι συνδεδεμένες λίστες που θα δούμε στη συνέχεια καταχωρούνται σε απομακρυσμένες θέσεις μνήμης που διασυνδέονται με δείκτες, λύνοντας έτσι το πρόβλημα αυτό.

### 8.5.3 Συνδεδεμένες γραμμικές λίστες

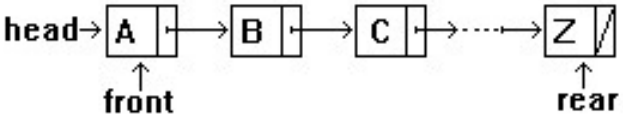
Μία συνδεδεμένη λίστα έχει την παρακάτω μορφή :



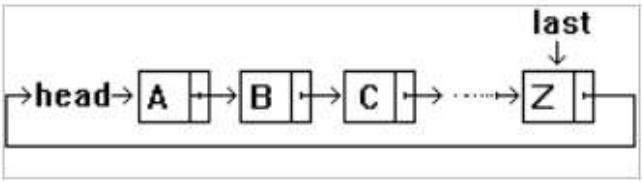

Σε μια γλώσσα προγραμματισμού, ο ορισμός του κάθε κόμβου της συνδεδεμένης γραμμικής λίστας θα περιλαμβάνει υποχρεωτικά δύο μέρη, το περιεχόμενο του κόμβου και μία μεταβλητή τύπου δείκτη που θα δείχνει στον επόμενο κόμβο. Θα πρέπει να οριστεί και μία επιπλέον μεταβλητή τύπου δείκτη (head) που θα δείχνει στην αρχή της λίστας.
Παρακάτω δείχνουμε σχηματικά την υλοποίηση μίας στοίβας σαν συνδεδεμένη λίστα.



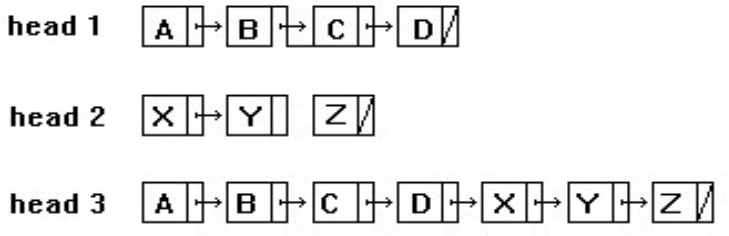

Μαζί με τον κόμβο, θα πρέπει να οριστεί και μία μεταβλητή τύπου δείκτη (top) που θα δείχνει στην κορυφή της στοίβας. Όταν top=nil η στοίβα είναι άδεια Σημειώνεται ότι όταν κάνουμε εισαγωγή (pop) μέσω νέου στοιχείου δεν είναι δυνατόν πάντα να γίνει έλεγχος για υπερχείλιση δηλαδή έλεγχος αν υπάρχει ελεύθερος χώρος στη μνήμη του Η/Υ.
Η υλοποίηση μιας ουράς σαν συνδεδεμένη λίστα φαίνεται σχηματικά παρακάτω :




Μαζί με τον κόμβο, θα πρέπει να οριστούν και δύο μεταβλητές front και rear τύπου δείκτη που θα δείχνουν στο εμπρός και το πίσω άκρο της στοίβας αντίστοιχα .
η παρακάτω δομή ονομάζεται κυκλική λίστα




Στη δομή αυτή είναι φανερό ότι αν ο δείκτης last δείχνει το τελευταίο στοιχείο της λίστας είναι εύκολο να βρούμε την αρχή της λίστας ως τον κόμβο που δείχνει ο δείκτης του κόμβου του τελευταίου στοιχείου.
Οι συνδεδεμένες λίστες μας δίνουν γενικά ευκολία στο χειρισμό τους, όπως στην εισαγωγή και εξαγωγή στοιχείων σε οποιοδήποτε σημείο τους.
Παρακάτω βλέπουμε σχηματικά τη λειτουργία της συνένωσης δύο λιστών σε μία ενιαία λίστα.




Διπλά συνδεδεμένη λίστα είναι μία λίστα την οποία επιτρέπεται να “διασχίσουμε” και προς τις δύο κατευθύνσεις. Για το σκοπό αυτό σε κάθε κόμβο χρησιμοποιούνται δύο δείκτες (π.χ. rightnxt, leftnxt), όπου ο ένας δείχνει τον επόμενο προς τα “δεξιά” και ο άλλος τον επόμενο προς τα “αριστερά” κόμβο της λίστας.

### 8.6 Ταξινόμηση πινάκων

Με τον όρο ταξινόμηση (sorting) εννοούμε την μετάθεση γενικά ενός συνόλου δεδομένων ώστε να τοποθετηθεί κατά αύξουσα ή φθίνουσα σειρά μεγέθους του (πρωτεύοντος) κλειδιού των κόμβων. Ένας βασικός σκοπός της ταξινόμησης είναι να διευκολύνει το έργο της αναζήτησης μέσα σε ένα σύνολο κόμβων.

Για την ταξινόμηση έχουν αναπτυχθεί πολλοί αλγόριθμοι. Βασικά κριτήρια για την μέτρηση της αποδοτικότητας των αλγορίθμων που χρησιμοποιούνται στην περίπτωση αυτή, είναι το πόσες συγκρίσεις κλειδιών χρειάζεται ο αλγόριθμος για να κάνει την ταξινόμηση και ο αριθμός μετακινήσεων στοιχείων (κόμβων) που απαιτούνται.

Οι πιο γνωστοί αλγόριθμοι ταξινόμησης είναι : ταξινόμηση με απευθείας παρεμβολή, ταξινόμηση με απ’ευθείας επιλογή και ταξινόμηση με απ’ευθείας αντιμετάθεση.

# ΚΕΦΑΛΑΙΟ 9. ΟΡΓΑΝΩΣΗ ΑΡΧΕΙΩΝ

## 9.1 Αρχεία και εγγραφές

‘Οπως είδαμε και στο κεφάλαιο 6.4 αρχείο (file) είναι γενικά ένα σύνολο πληροφοριών που καταχωρούνται συνήθως σε κάποιο μέσο αποθήκευσης σύμφωνα με κάποια φυσική και λογική οργάνωση. Ακόμα ο τρόπος οργάνωσης των αρχείων (δηλαδή των δεδομένων μέσα στο αρχείο) εξαρτάται από το μέσο αποθήκευσης (μαγνητική ταινία, μαγνητικό τύμπανο, μαγνητικός δίσκος κλπ). Εδώ πρέπει να σημειωθεί ότι αναφερόμαστε βασικά στα αρχεία εφαρμογής (application files). Εκτός από αυτά υπάρχουν και τα αρχεία συστήματος (system files) που παραμένουν μόνιμα στον υπολογιστή.

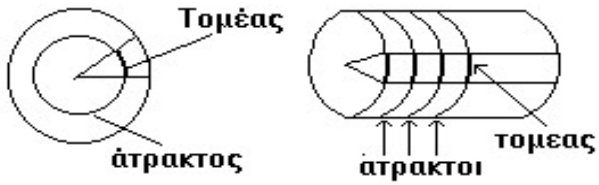
Είδαμε ακόμη ότι ένα αρχείο είναι ένα σύνολο εγγραφών (records). Για παράδειγμα αν έχομε ένα αρχείο υπαλλήλων, το σύνολο των στοιχείων που αφορούν έναν υπάλληλο θα αποτελούν μία εγγραφή του αρχείου. Κάθε εγγραφή αποτελείται από διάφορα μέρη που ονομάζονται πεδία (fields) π.χ. στο αρχείο υπαλλήλων που αναφέρθηκε, η εγγραφή ενός υπαλλήλου θα αποτελείται από πεδία που μπορούν να είναι, για παράδειγμα : ονοματεπώνυμο υπαλλήλου, κωδικός υπαλλήλου, διεύθυνση κατοικίας, ημερομηνία γέννησης, τηλέφωνο, email, μισθός κλπ. Κάθε πεδίο της κάθε εγγραφής θα είναι στην πραγματικότητα μία σειρά από bytes στη μνήμη (Κύρια ή βοηθητική) του υπολογιστή. Εδώ πρέπει να σημειώσουμε οτι η κάθε πληροφορία που μας ενδιαφέρει , πρέπει να καταχωρείται σε ξεχωριστό πεδίο διότι αυτό θα μας βοηθήσει στην οργάνωση και επεξεργασία του αρχείου, που είναι πάντα το ζητούμενο. Αν για παράδειγμα στο πεδίο της διεύθυνσης ενσωματώσουμε και την πόλη, δεν θα είναι εύκολο αν χρειαστεί να εντοπίσουμε τους υπαλλήλους σε μία συγκεκριμένη πόλη.

Η διαχείριση και επεξεργασία αρχείων περιλαμβάνει ένα σύνολο εργασιών πάνω σε ένα αρχείο για να επιτύχουμε ένα επιθυμητό αποτέλεσμα. Κάποιες βασικές τέτοιες εργασίες είναι οι παρακάτω :

- προσθήκη (add) εγγραφής στο αρχείο (προσθήκη νέου υπαλλήλου, στο ανωτέρω παράδειγμα)
- ταξινόμηση (sort) των εγγραφών με έναν συγκεκριμένο κανόνα (πχ ταξινόμηση των υπαλλήλων με αλφαβητική σειρά)
- αναζήτηση (search) κάποιων εγγραφών με κάποιο κριτήριο (πχ υπάλληλοι που λαμβάνουν μισθό πάνω από κάποιο ποσό)
- επεξεργασία (edit) μιας συγκεκριμένης εγγραφής (της “καρτέλας” κάποιου υπαλλήλου)
- διαγραφή (delete) κάποιων εγγραφών (κάποιων υπαλλήλων)
- εκτύπωση (print) του συνόλου ή κάποιου υποσυνόλου των εγγραφών, με καθορισμένη μορφή (πχ το σύνολο ή κάποιο υποσύνολο των υπαλλήλων με ορισμένα πεδία, π.χ ονοματεπώνυμο, κωδικός και μισθός)

Στην επεξεργασία αρχείων συναντάμε την έννοια του κλειδιού. Το κλειδί (key) είναι τμήμα της εγγραφής, που την προσδιορίζει και αποτελεί πολλές φορές τη βάση οργανώσεως ενός αρχείου. Π.χ. ο κωδικός υπαλλήλου μπορεί να αποτελεί το κλειδί για το παραπάνω αρχείο υπαλλήλων του οποίου οι εγγραφές μπορεί να είναι ταξινομημένες κατά αύξουσα σειρά του κωδικού αυτού. Σ'αυτήν την περίπτωση λέγεται και πρωτεύον κλειδί (primary key) ενώ μπορεί να υπάρχουν και δευτερεύοντα κλειδιά (secondary keys).

'Οπως είδαμε στο κεφάλαιο 5 οι πληροφορίες (άρα και τα αρχεία) αποθηκεύονται με μαγνητικό τρόπο πάνω στην επιφάνεια των μαγνητικών δίσκων. Η μαγνητική επιφάνεια διαιρείται σε ομόκεντρους κύκλους που λέγονται ίχνη ή άτρακτοι (tracks), και αυτοί χωρίζονται σε τομείς (sectors) . Στο μαγνητικό τύμπανο (magnetic drum), οι άτρακτοι βρίσκονται σε παράλληλους κύκλους κατά μήκος της κυλινδρικής επιφάνειας του κυλίνδρου.




Μαγνητικός δίσκος Μαγνητικό τύμπανο

Η διακίνηση πληροφοριών μεταξύ του Η/Υ και μιας περιφερειακής μονάδας μνήμης γίνεται με ολόκληρα σύνολα bytes (ή λέξεων). Μια τέτοια μονάδα μεταφοράς πληροφοριών λέγεται φυσική εγγραφή (physical record ή block) σε αντιδιαστολη με την λογική εγγραφή (logical record) που είναι ουσιαστικά η εγγραφή όπως την ορίσαμε προηγούμενα, δηλαδή όπως την βλέπει ο χρήστης (με τα πεδία που την αποτελούν) ανεξάρτητα με το πως αυτή καταχωρείται στη μονάδα μνήμης. Μια φυσική εγγραφή μπορεί να περιλαμβάνει μία ή περισσότερες λογικές εγγραφές, αλλά είναι επίσης δυνατόν μία λογική εγγραφή να καταλαμβάνει περισσότερες από μία φυσικές εγγραφές.

'Οταν ένα αρχείο καταχωρείται σε κάποιο μαγνητικό μέσον, τότε μεταξύ των φυσικών εγγραφών παρεμβάλλονται κενά που ονομάζονται Inter Block Gaps (IGB). Υπάρχει όμως η δυνατότητα περισσότερες από μία λογικές εγγραφές να ομαδοποιούνται σε μία φυσική εγγραφή, που σ'αυτήν την περίπτωση θα έχει μεγαλύτερο μήκος, και μετά από αυτή να ακολουθεί IBG. Με αυτόν τον τρόπο γίνεται καλλίτερη αξιοποίηση του μαγνητικού μέσου αποθήκευσης. Στην πρώτη περίπτωση λέμε ότι έχομε μη ομαδοποιημένες εγγραφές ενώ στη δεύτερη ομαδοποιημένες

| | | | | | | |
|---|---|---|---|---|---|---|
| IBG | REC.1 | IBG | REC.2 | IBG | REC.3 | IBG |

Μη ομαδοποιημένες εγγραφές

| IBG | REC.1 | REC.2 | REC.3 | IBG | REC.4 | REC.5 | REC.6 | IBG |
|---|---|---|---|---|---|---|---|---|

Ομαδοποιημένες εγγραφές.

‘Ενα αρχείο περιλαμβάνει εκτός από τις εγγραφές πεδίων (field records) που περιέχουν τις κυρίως πληροφορίες και την εγγραφή ετικέτας (label record) που βρίσκεται στην αρχή του και περιέχει γενικές πληροφορίες για το αρχείο (όνομα, ημερομηνία, δημιουργίας του, μέγεθος κλπ). Στο τέλος του αρχείου υπάρχει συνήθως και η “ρυμουλκούμενη” εγγραφή (trailer record) που η λειτουργία της είναι ανάλογη με εκείνη της εγγραφής ετικέτας.

### 9.2 Είδη αρχείων

Τα αρχεία ανάλογα με το είδος των πληροφοριών που περιέχουν και τη χρήση τους χωρίζονται στις παρακάτω βασικές κατηγορίες .

α) Κύρια αρχεία (master files). Είναι το βασικό αρχείο μιας εφαρμογής που περιέχει τις κύριες εγγραφές με τις πληροφορίες που χρειάζονται. Τέτοια αρχεία είναι π.χ. αρχεία πελατών, αρχεία υπαλλήλων, αρχεία υλικών αποθήκης κλπ. Συνήθως στα αρχεία αυτά γίνεται τακτική ενημέρωση (update) ώστε να περιέχουν κάθε στιγμή τις πιο πρόσφατες πληροφορίες.

β) Αρχεία μεταβολών (transaction files). Σε ένα τέτοιο αρχείο καταχωρούνται οι αλλαγές που προκύπτουν στα δεδομένα κάποιου κύριου αρχείου, για κάποιο χρονικό διάστημα (π.χ. στη διάρκεια της ημέρας). Το αρχείο αυτό χρησιμοποιείται στη συνέχεια για ενημέρωση του κυρίως αρχείου.

γ) Εφεδρικά αρχεία (backup files). Είναι αντίγραφα κάποιου κυρίως αρχείου που φυλάγονται για ασφάλεια, σε περίπτωση καταστροφής του κυρίως αρχείου.

δ) Αρχεία δεικτών (index files). Τα αρχεία αυτά περιέχουν δείκτες/διευθύνσεις προς εγγραφές ή γενικότερα πληροφορίες που βρίσκονται σε άλλα αρχεία. Αντιστοιχούν στους γνωστούς μας πίνακες περιεχομένων.

ε) Αρχεία αναφοράς (reference files). Τα αρχεία αυτά περιέχουν κάποια στοιχεία όπου χρειαζόμαστε συχνά να αναφερόμαστε και είναι συνήθως σε κάπως συνοπτική μορφή.

### 9.3. Μέθοδοι οργάνωσης και αναζήτησης αρχείων

‘Οταν λέμε οργάνωση αρχείων εννοούμε την οργάνωση και λογικό συσχετισμό μεταξύ των εγγραφών του, ώστε να γίνεται όσο το δυνατόν πιο εύκολα ο εντοπισμός και η ανάκτηση (retrieval) αυτών, για να γίνει πιθανώς στη συνέχεια ενημέρωση (updating) του αρχείου. Είναι σημαντικό να βρούμε τον καλλίτερο δυνατό τρόπο οργάνωσης του αρχείου ώστε να επιτύχουμε γρήγορη προσπέλαση, εύκολη ενημέρωση, εξοικονόμηση χώρου, ασφάλεια κλπ. Πρέπει επίσης να εξετάσουμε τον τρόπο επεξεργασίας που πρόκειται να γίνει στο αρχείο (on

line, batch κλπ.) και το μέσο για την αποθήκευση του αρχείου που θα χρησιμοποιηθεί. Σε σχέση με τις μεθόδους οργάνωσης των αρχείων είναι και οι μέθοδοι αναζήτησης (searching) για την ανεύρεση κάποιου κλειδιού μέσα στο αρχείο. Στη συνέχεια θα εξετάσουμε μερικές από τις πιο γνωστές μεθόδους οργάνωσης και αναζήτησης αρχείων.

### 9.3.1 Σειριακή (sequential) οργάνωση

Σύμφωνα με τη σειριακή οργάνωση, οι εγγραφές του αρχείου είναι καταχωρημένες η μία μετά την άλλη, σύμφωνα με την τιμή κάποιου κλειδιού της εγγραφής (π.χ. κωδικός αριθμός) που είναι συνήθως κάποιο πεδίο της. Για να διαβαστεί κάποια εγγραφή πρέπει προηγουμένως να διαβαστούν όλες οι άλλες εγγραφές που προηγούνται, με τη σειρά. Ακόμη, όταν πρέπει να προστεθούν νέες εγγραφές και να παρεμβληθούν στο αρχείο σύμφωνα με τον κωδικό, του κλειδιού τους ή να διαγραφούν κάποιες εγγραφές, πρέπει να δημιουργηθεί ένα νέο αρχείο, με βάση τον προηγούμενο και τις απαιτούμενες προσθήκες/διαγραφές εγγραφών. Για τους λόγους αυτούς ή σειριακή οργάνωση δεν είναι αποδοτική όταν η εφαρμογή που δουλεύουμε απαιτεί συχνή επεξεργασία τυχαίων, μεμονωμένων εγγραφών. Είναι κατάλληλη όταν έχομε επεξεργασία ολόκληρου του αρχείου ή μεγάλου πλήθους εγγραφών του αρχείου.

Η σειριακή οργάνωση χρησιμοποιείται συνήθως σε μονάδες μαγνητικών ταινιών, αν και μπορεί να χρησιμοποιηθεί και σε συσκευές άμεσης προσπέλασης (όπως μαγνητικούς δίσκους). Και σ'αυτήν την περίπτωση οι διαδοχικές εγγραφές καταχωρούνται σε διαδοχικές φυσικές διευθύνσεις (διαδοχικές ατράκτους και τομείς, διαδοχικούς δίσκους κλπ). Στην περίπτωση αυτή η σειριακή οργάνωση χρησιμοποιείται περισσότερο για πίνακες, ευρετήρια κλπ. και όχι για κύρια αρχεία.

Μπορούμε να εκμεταλλευτούμε το γεγονός ότι οι εγγραφές του αρχείου είναι ταξινομημένες (π.χ. κατά αύξουσα σειρά) για να επιταχύνουμε τη διαδικασία αναζήτησης κάποιου κλειδιού. Η πιο συνηθισμένη μέθοδος αυτής της κατηγορίας είναι η δυαδική αναζήτηση. Σύμφωνα με τη μέθοδο αυτή συγκρίνουμε το ζητούμενο κλειδί με την τιμή του κλειδιού της μεσαίας εγγραφής του αρχείου (ή με τη μέση τιμή κλειδιών του αρχείου). Αν το ζητούμενο κλειδί είναι μεγαλύτερο εξαιρούμε το πρώτο μισό του αρχείου, ενώ αν είναι μικρότερο εξαιρούμε το δεύτερο μισό του αρχείου. Επαναλαμβάνουμε τη διαδικασία με το μισό αρχείο που απέμεινε κοκ. μέχρι να βρούμε το κλειδί ή μέχρι να διαπιστώσουμε ότι το κλειδί δεν υπάρχει στο αρχείο.

Υπάρχει και η μη ταξινομημένη σειριακή (serial) οργάνωση, όπου οι εγγραφές του αρχείου καταχωρούνται η μία μετά την άλλη χωρίς κάποια ταξινόμηση. Η οργάνωση αυτή χρησιμοποιείται π.χ. όταν συγκεντρώνουμε τις νέες εγγραφές που πρόκειται να εισαχθούν σε κάποιο αρχείο. Συνήθως πριν γίνει επεξεργασία, οι εγγραφές ενός τέτοιου αρχείου ταξινομούνται κατάλληλα με βάση κάποιο κλειδί.

### 9.3.2 Σειριακή οργάνωση με δείκτη (indexed sequential)

Η οργάνωση αυτή προϋποθέτει ότι οι εγγραφές του αρχείου είναι καταχωρημένες σε διαδοχικές θέσεις φυσικής μνήμης και σύμφωνα με τη σειρά που καθορίζει το κλειδί κάθε εγγραφής. Μ'αυτόν τον τρόπο υπάρχει δυνατότητα για σειριακή προσπέλαση του αρχείου, ταυτόχρονα όμως υπάρχει και η δυνατότητα για γρήγορη (απ'ευθείας) προσπέλαση σε κάποια εγγραφή του αρχείου, με βάση το κλειδί. Αυτό επιτυγχάνεται με τη χρήση ενός πίνακα δεικτών, που συσχετίζει το κλειδί κάθε εγγραφής με τη φυσική διεύθυνση της εγγραφής στο μέσο αποθήκευσης (που θα πρέπει να είναι συσκευή άμεσης προσπέλασης). Για παράδειγμα, υποθέτουμε ότι έχομε συσκευή άμεσης προσπέλασης που αποτελείται από κυλίνδρους, οι οποίοι χωρίζονται σε ατράκτους. Κάθε κύλινδρος μπορεί να περιέχει μέχρι 4 ατράκτους (0, 1, 2, 3) ενώ σε κάθε άτρακτο καταχωρούνται μέχρι 3 εγγραφές. Έστω έχομε αρχείο με 17 εγγραφές με κλειδιά 01, 02, 04, 05, 06, 08, 11, 12, 13, 14, 15, 16, 17, 18, 23, 25, 26.

Το αρχείο αυτό χρειάζεται 2 κυλίνδρους για να καταχωρηθεί, έστω τους κυλίνδρους με αριθμούς 15, 16. Η φυσική διεύθυνση αποτελείται από τον αριθμό κυλίνδρου ακολουθούμενο από τον αριθμό ατράκτου.

Η καταχώρηση του αρχείου θα γίνει ως εξής :

| Φυσική διεύθυνση | Κλειδιά των εγγραφών που περιέχονται |
|---|---|
| 150 | 01, 02, 04 |
| 151 | 05, 06, 08 |
| 152 | 11, 12, 13 |
| 153 | 14, 15, 16 |
| 160 | 17, 18, 23 |
| 161 | 25, 26 |

Ο πίνακας δεικτών περιλαμβάνει, αριθμούς κυλίνδρων, αριθμούς ατράκτων, και για κάθε άτρακτο κάθε κυλίνδρου το μεγαλύτερο κλειδί που περιέχεται στην άτρακτο αυτή. Για το παραπάνω παράδειγμα ο πίνακας δεικτών θα είναι :

| Αριθμός κυλίνδρου | Αριθμός ατράκτου | μεγαλύτερο κλειδί |
|---|---|---|
| 15 | 0 | 04 |
| 15 | 1 | 08 |
| 15 | 2 | 13 |
| 15 | 3 | 16 |
| 16 | 0 | 23 |
| 16 | 1 | 26 |

Η πρώτη και η τρίτη στήλη του πίνακα αποτελούν το δείκτη κυλίνδρων, ενώ η δεύτερη και η τρίτη στήλη, τον δείκτη ατράκτων.

‘Οταν γίνεται αναζήτηση για να βρεθεί κάποιο κλειδί (π.χ. το 14) πρώτα από τον δείκτη κυλίνδρων βρίσκουμε ότι αυτό θα βρίσκεται στον κύλινδρο 15. Στη συνέχεια από τον δείκτη ατράκτων προκύπτει ότι το ζητούμενο κλειδί βρίσκεται στην άτρακτο 3 (επειδή 13<14<16). Τέλος ολόκληρη η άτρακτος 153 μεταφέρεται στην Κεντρική Μνήμη και εκεί με σειριακή προσπέλαση βρίσκεται η εγγραφή με το ζητούμενο κλειδί.

Για να αντιμετωπιστεί το πρόβλημα με την εισαγωγή νέων εγγραφών στο αρχείο, υπάρχει σε κάθε κύλινδρο μία άτρακτος υπερχείλισης, όπου τοποθετούνται οι νέες εγγραφές, και για την περίπτωση που και αυτές θα γεμίσουν, υπάρχει και κύλινδρος υπερχείλισης. ‘Οταν μία εγγραφή διαγράφεται, συνήθως δεν απομακρύνεται από το αρχείο, αλλά τοποθετείται σ’αυτήν κάποια ειδική σήμανση που δείχνει ότι είναι άκυρη.

### 9.3.3 Απ’ευθείας (direct ή random) οργάνωση

Και η μέθοδος αυτή χρησιμοποιείται σε συσκευές άμεσης προσπέλασης. Η οργάνωση αυτή επιτρέπει την απ’ευθείας προσπέλαση μίας εγγραφής του αρχείου (με βάση το κλειδί της) και αυτό επιτυγχάνεται με κάποιον μηχανισμό μέσω του οποίου μεταβαίνουμε από το κλειδί μιας εγγραφής στη φυσική διεύθυνση όπου βρίσκεται καταχωρημένη. Οι εγγραφές σ’αυτήν την περίπτωση μπορεί να μην είναι καταχωρημένες με κάποια σειρά. Σ’αυτήν τη κατηγορία οργάνωσης διακρίνουμε τρεις υποπεριπτώσεις.

α) ‘Αμεση (direct) οργάνωση. Το κλειδί που χρησιμοποιείται από τον χρήστη είναι η ίδια η φυσική διεύθυνση της εγγραφής.

β) Οργάνωση με τη βοήθεια λεξικού (dictionary). Υπάρχει κάποιο λεξικό που συσχετίζει τα κλειδιά των εγγραφών με τις φυσικές διευθύνσεις των αντίστοιχων εγγραφών. Επειδή όταν το αρχείο μεγαλώσει πολύ, μεγαλώνει και το λεξικό, και η αναζήτηση ενός κλειδιού σ΄αυτό γίνεται χρονοβόρα, συνήθως το λεξικό αυτό ταξινομείται σύμφωνα με το κλειδί των εγγραφών και εφαρμόζεται δυαδική αναζήτηση.

γ) Οργάνωση με μετασχηματισμό του κλειδιού (hashing)

Σύμφωνα με τη μέθοδο αυτή, χρησιμοποιείται μία συνάρτηση που αντιστοιχεί στο κλειδί μιας εγγραφής κάποιον αριθμό, που είναι η διεύθυνση (θέση) στην οποία καταχωρείται η εγγραφή μέσα στο αρχείο (bucket), π.χ. διαιρούμε το κλειδί με κάποιον πρώτο αριθμό και κρατάμε το υπόλοιπο. Με τη μέθοδο αυτή υπάρχει πάντα περίπτωση σύγκρουσης (collision). Δηλαδή από δύο διαφορετικά κλειδιά να προκύψει η ίδια διεύθυνση ή θέση (bucket) μέσα στο αρχείο. Τότε από τη διεύθυνση που προέκυψε, αρχίζει αναζήτηση στο αρχείο σειριακά, μέχρι να βρεθεί το ζητούμενο κλειδί. Συνήθως τα αρχεία που δημιουργούνται για να εφαρμοστεί η μέθοδος αυτή, καταλαμβάνουν τουλάχιστον 50% περισσότερο χώρο από όσο απαιτούν οι εγγραφές αυτές καθ’αυτές, για να ελαχιστοποιείται η πιθανότητα σύγκρουσης. Σε

αντιστάθμισμα η μέθοδος αυτή οργάνωσης επιτυγχάνει πολύ μικρούς χρόνους προσπέλασης.

### 9.3.4 Αναζήτηση με βάση δευτερεύοντα κλειδιά

Μέχρι τώρα εξετάσαμε το πρόβλημα της αναζήτησης εγγραφών σ'ένα αρχείο, με βάση κάποιο πρωτεύον κλειδί (primary key) που είναι συνήθως κάποιο πεδίο της εγγραφής. Πολλές φορές παρουσιάζονται όμως προβλήματα όπου έχομε αναζήτηση με βάση κάποιο άλλο πεδίο. 'Εστω π.χ. ότι έχομε ένα αρχείο υπαλλήλων μίας εταιρείας, και κάθε εγγραφή περιλαμβάνει τα πεδία : Κωδικός υπαλλήλου, Ονοματεπώνυμο, Φύλο, Ειδικότητα, Ηλικία, Μισθός. Ο κωδικός θα είναι λογικά το πρωτεύον κλειδί του αρχείου, μπορεί όμως να θέλουμε να βρούμε π.χ. όλους τους υπαλλήλους που:

1) Είναι άνδρες (απλή ερώτηση).
2) Είναι μεγαλύτεροι των 40 ετών (ερώτηση διαστήματος).
3) 'Εχουν ειδικότητα "προγραμματιστής" και είναι μικρότεροι των 30 ετών (λογική ερώτηση).
4) 'Εχουν μισθό μεγαλύτερο από το μέσο μισθό όλων των υπαλλήλων (συναρτησιακή ερώτηση).

'Ολες αυτές οι απαιτήσεις, εκφράζονται συνήθως με κάποια ειδική γλώσσα ερωταποκρίσεων (query language). Για να γίνει η επεξεργασία του αρχείου με βάση ερωτήσεις αυτής της μορφής, δημιουργείται ένας αριθμός από δευτερεύοντες δείκτες ή κλειδιά (secondary indexes), όπου κάθε δείκτης αντιστοιχεί σε κάποιο δευτερεύον πεδίο (Ειδικότητα, Ηλικία, Μισθός κλπ). Υπάρχουν διάφορες μέθοδοι για την υλοποίηση των δευτερευόντων δεικτών και εν συνεχεία την αναζήτηση εγγραφών μέσω αυτών, οι πιο γνωστοί από τους οποίους είναι η μέθοδος των αντιστραμμένων λεξικών ή αρχείων και η μέθοδος των πολλαπλών λιστών.

# ΚΕΦΑΛΑΙΟ 10. ΕΦΑΡΜΟΓΕΣ ΤΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ – ΕΙΔΙΚΑ ΘΕΜΑΤΑ

## 10.1 Εφαρμογές των Η/Υ.

Στο Κεφάλαιο 1 είδαμε ήδη εν συντομία περιοχές όπου οι Η/Υ γνωρίζουν ευρεία χρήση, αν και όπως επίσης αναφέρθηκε, οι Η/Υ, σε συνδυασμό με τη διάδοση του Διαδικτύου έχουν διεισδύσει πρακτικά σε όλους τους τομείς της σύγχρονης ζωής, ενώ σε πολλές περιπτώσεις η χρήση τους έχει καταστεί απαραίτητη.

Εδώ θα δούμε περισσότερες πληροφορίες σε σχέση με το πεδίο εφαρμογής των Η/Υ και τις κατηγορίες των εφαρμογών τους.

Εφαρμογές στο χώρο των επιχειρήσεων

Παραδείγματα

α) Εφαρμογές στο χώρο του γραφείου (office automation). Λογισμικά (software packages/suites) της μορφής αυτής περιλαμβάνουν τα πολύ γνωστά προγράμματα επεξεργασίας κειμένου (word processors), λογιστικά φύλλα (spreadsheets), λογισμικό παρουσιάσεων (presentation software), ηλεκτρονικού ταχυδρομείου και επικοινωνίας (email/communication) κ.α.

β) Τα πακέτα λογισμικού που αναφέρονται ως ERP (Enterprise Resource Planning), όπως SAP, ORACLE cloud, SAGE κ.α. που έχουν αναπτυχθεί για να καλύψουν όλο τα φάσμα των δραστηριοτήτων μιας (μεγάλης κατά κανόνα) επιχείρησης , όπως

- Οικονομικές συναλλαγές – Λογιστήριο – Κοστολόγηση -Οικονομικές αναφορές
- Προμήθειες- Αποθήκες-εφοδιαστική αλυσίδα
- Διαχείριση παραγγελιών – τιμολόγηση
- Διαχείριση σχέσεων με πελάτες και πωλήσεις (CRM : Customer Relationship Management)
- Προγραμματισμός και διαχείριση παραγωγής
- Διαχείριση προσωπικού – μισθοδοσία- αξιολόγηση

γ) Ηλεκτρονικό εμπόριο. Εφαρμογές και πλατφόρμες για διαφήμιση, προβολή (digital marketing) και αγοροπωλησία προιόντων και υπηρεσιών από απόσταση (μέσω του Διαδικτύου).

δ) Εφαρμογές για τραπεζικά ιδρύματα. Διαχείριση χαρτοφυλακίου πελατών, ιδιωτών και επιχειρήσεων (private/business) π.χ. τραπεζικές συναλλαγές και κίνηση λογαριασμών, δάνεια, κάρτες, ασφαλιστικά και επενδυτικά προιόντα, συνάλλαγμα. Ηλεκτρονικές συναλλαγές μέσω υπολογιστή ή κινητού τηλεφώνου (e-banking/mobile banking) κ.α.

ε) Εφαρμογές για οργανισμούς και εταιρίες κοινής ωφέλειας, όπως έκδοση, διαχείριση και πληρωμή λογαριασμών ηλεκτρικού ρεύματος, τηλεπικοινωνιών, ύδρευσης/αποχέτευσης κ.α.

στ) Εφαρμογές για δημόσιες υπηρεσίες, Δυνατότητα πραγματοποίησης πλήθους συναλλαγών με φορείς και Οργανισμούς του Δημοσίου, όπως : υπολογισμός και έκδοση

παραστατικών για την πληρωμή φόρου εισοδήματος και λοιπών φόρων, ασφαλιστικών εισφορών, έκδοση πιστοποιητικών, κτηματολόγιο, αποτελέσματα εκλογών, αποτελέσματα εισαγωγικών και άλλων εξετάσεων εκπαιδευτικών ιδρυμάτων κ.α

Επιστημονικές και άλλες εφαρμογές

α) Επίλυση μαθηματικών – υπολογιστικών προβλημάτων (π.χ. στατιστικής, αριθμητικής ανάλυσης κλπ) μετεωρολογία, γραμμικός προγραμματισμός κλπ.

β) Εξομοίωση (simulation) συστήματος. Μαθηματική ή συμβολική αναπαράσταση ενός συστήματος στον Η/Υ.

γ) Ο υπολογιστής στη σχεδίαση, παραγωγή και διοίκηση. Εφαρμογές και προγράμματα αυτής της κατηγορίας αναφέρονται συνήθως ως :

- CAD (computer aided design)
- CAM (computer aided manufacturing)
- CIM (computer integrated manufacturing)
- MIS (Management Information Systems)

δ) Διαχείριση και χρονοπρογραμματισμός έργου (project management). Τα προγράμματα αυτά χρησιμεύουν στο σχεδιασμό και την ανάλυση των διαφόρων φάσεων μιας διαδικασίας ή εκτέλεσης ενός έργου.

ε) Εφαρμογές ελέγχου και διαχείρισης συστημάτων οδικής κυκλοφορίας, ενέργειας και περιβάλλοντος όπως : ρύθμιση οδικής κυκλοφορίας (ρύθμιση φωτεινών σηματοδοτών, διασύνδεση καμερών κυκλοφορίας), διαχείριση στόλου οχημάτων (fleet mamagement), “έξυπνες” στάσεις μέσων μαζικής μεταφοράς, συστήματα αυτόματης πλοήγησης οχημάτων, διαχείριση συστημάτων ενέργειας (π.χ. φωτισμού, κτηρίων), διαχείριση απορριμάτων, μετρήσεις και έλεγχος περιβαλλοντικών συνθηκών κλπ. Οι εφαρμογές αυτού του είδους αναφέρονται συχνά ως εφαρμογές “Έξυπνης Πόλης” (smart cities).

στ) Γεωγραφικά Συστήματα Πληροφοριών (GIS : Geographical Information Systems). Εξειδικευμένα προγράμματα για την αναπαράσταση και επεξεργασία γεωχωρικών δεδομένων

ζ) Εφαρμογές σε δίκτυα μεταφορών, δημόσιων ή ιδιωτικών, π.χ έλεγχος και διαχείριση σιδηροδρόμων, συστήματα εναέριας κυκλοφορίας.

η) Αυτοματοποίηση και έλεγχος παραγωγής, (συστήματα SCADA, PLC κλπ) – ρομποτική

θ) Εφαρμογές στην Ιατρική. Επεξεργασία βιολογικών σημάτων (π.χ. ηλεκτροεγκεφαλογρά-φημα : ΗΕΓ, ηλεκτροκαρδιογράφημα : ΗΚΓ, ηλεκτρομυογράφημα : ΕΜΓ), χρήση Η/Υ στη διάγνωση ασθενειών με βάση τα συμπτώματα (Έμπειρα Συστήματα/Τεχνητή Νοημοσύνη), τηλειατρική. Δυνατότητα για διάγνωση αλλά και θεραπεία ασθενειών από απόσταση (συνταγογράφηση φαρμάκων, λήψη και εκτίμηση αποτελεσμάτων εξετάσεων κ.α.).

ι) Τηλεεργασία. Υπάρχει πλέον δυνατότητα να εργάζεται κάποιος ευρισκόμενος σε απόσταση από το χώρο εργασίας (συνήθως γραφεία μιας εταιρίας), π.χ. από το σπίτι του, έχοντας

πλήρη πρόσβαση στο υπολογιστικό σύστημα της εταιρίας, καθώς και δυνατότητες τηλεδιάσκεψης.

ια) Συστήματα ασφαλείας και ελέγχου, εγκαταστάσεων και συστημάτων κάθε είδους.

ιβ) Στον τομέα της άμυνας υπάρχει πλήθος συστημάτων και εφαρμογών για τον έλεγχο και καθοδήγηση διαφόρων τύπων οπλικών συστημάτων, χερσαίων, θαλάσσιων, εναερίων, βλημάτων, πυραύλων καθώς και συστήματα ανίχνευσης και προειδοποίησης (ραντάρ), ηλεκτρονικού πολέμου κ.α.

ιγ) Επεξεργασία σημάτων, εικόνων, video. Εφαρμογές διαχείρισης και επεξεργασίας αρχείων πολυμέσων (multimedia)

ιδ) Μετάφραση γλωσσών.

ιε) Εφαρμογές στις τέχνες (μουσική, ζωγραφική κλπ) και στην ψυχαγωγία (multimedia, gaming κ.λ.π.)

Οι Η/Υ σε συνεργασία με το Διαδίκτυο έχουν εισβάλει πλέον και σε σχεδόν όλους τους τομείς της καθημερινής ζωής, επικοινωνία, ενημέρωση, διασκέδαση κλπ. Πολύ σημαντικό ρόλο σε αυτό έχει παίξει η εντυπωσιακή ανάπτυξη τις τελευταίες δεκαετίες των μέσων κοινωνικής δικτύωσης (social media).

### 13.2 Η πληροφορική στην εκπαίδευση.

Η χρήση των Η/Υ στην εκπαίδευση (e-learning), δηλαδή ο υπολογιστής όχι ως αντικείμενο αλλά ως μέσο διδασκαλίας, αναφέρεται συνήθως και ως CAL (Computer Aided Learning). Με τη βοήθεια ενός συστήματος e-learning ο υπολογιστής χρησιμοποιείται ως μέσον για την παρουσίαση εκπαιδευτικής ύλης. Η χρήση συστημάτων e-learning είναι αποδεκτή, αλλά μόνο σε ορισμένες περιπτώσεις έχει αποδειχθεί αποδοτικώτερη από τις κλασσικές μεθόδους διδασκαλίας (παρακολούθηση διαλέξεων, διάβασμα κλπ). Για την ανάπτυξη συστημάτων e-learning χρησιμοποιούνται τεχνολογίες όπως : εξ'αποστάσεως εκπαίδευση (Distance Learning), Βάσεις Δεδομένων, προσομοίωση, Έμπειρα Συστήματα/τεχνητή νοημοσύνη, multimedia, κλπ. Στον ευρύτερο πεδίο της εκπαίδευσης με χρήση Η/Υ ανήκουν τα Συστήματα Διαχείρισης Μάθησης (LMS: Learning Management Systems), όπως οι πλατφόρμες Moodle και Canvas, πλατφόρμες μαθημάτων online (π.χ. edX, Coursera), καθώς και πλατφόρμες που χρησιμοποιούνται για πιο εξειδικευμένα αντικείμενα εκπαίδευσης (π.χ. ξένες γλώσσες, τέχνες).

Χαρακτηριστικά <u>πλεονεκτήματα των συστημάτων CAL</u>.

1) Η τεχνική CAL είναι ευέλικτη (flexible) και μπορεί να προσαρμοστεί εύκολα ανά περίπτωση. Τα συστήματα CAL έχουν επίσης τη δυνατότητα να συνδέονται ανά περίπτωση με συσκευές video ή άλλα συστήματα παρουσίασης- επίδειξης.

2) Με τα συστήματα CAL η γνώση παρουσιάζεται στο μαθητή υπό μορφή ενοτήτων. Το μέγεθος και το επίπεδο κάθε ενότητας εξαρτάται από το επίπεδο των μαθητών, την εκπαιδευτική μέθοδο κλπ.

3) Υπάρχει αλληλεπίδραση μεταξύ μαθητή και συστήματος (ο μαθητής υποβάλλει ερωτήσεις και απαντά σε ερωτήσεις) και μάλιστα σύμφωνα με τον προσωπικό ρυθμό του κάθε μαθητή, σε αντίθεση με τον λίγο – πολύ παθητικό ρόλο του μαθητή στις κλασσικές μεθόδους διδασκαλίας.

4) Ένα σύστημα CAL εξετάζει πιο εύκολα το βαθμό κατανόησης του αντικειμένου από το μαθητή (με ερωτήσεις tests κλπ) , καταγράφει πληροφορίες σε σχέση με την επίδοση των μαθητών και αποφασίζει για επιπλέον βοηθητικές ενέργειες ανά περίπτωση.

5) Ο μαθητής μπορεί να πληροφορηθεί άμεσα για την επίδοσή του σε κάποιο test ή ερώτηση που του υποβλήθηκε.

6) Έχει εκτιμηθεί ότι ένα καλό σύστημα CAL μπορεί να επιτύχει το ίδιο αποτέλεσμα με έναν καλό εκπαιδευτή σε λιγότερο χρόνο.

7) Η εκπαίδευση μπορεί να γίνει οποιαδήποτε χρονική στιγμή το σύστημα CAL είναι προσπελάσιμο.

8) Επιτυγχάνεται τυποποίηση και ομοιομορφία στο εκπαιδευτικό υλικό.

9) Ένα σύστημα CAL απαλλάσσει τον εκπαιδευτή από διαδικασίες διδασκαλίας και βαθμολόγησης (αξιολόγησης) ώστε να μπορέσει να ασχοληθεί με άλλες δραστηριότητες (πχ να βοηθήσει τους πιο αδύνατους μαθητές).

<u>Μειονεκτήματα των συστημάτων CAL</u>

1) Κόστος κτήσης ή/και υλοποίησης-προσαρμογής-παραμετροποίησης του συστήματος CAL, κυρίως όσον αφορά το λογισμικό.

2) Δεν μπορεί να αποκατασταθεί πλήρως η ανθρώπινη επικοινωνία (δασκάλου-μαθητή) κυρίως λόγω περιορισμών από μονάδες εισόδου-εξόδου.

3) Δεν υπάρχουν κοινά αποδεκτά πρότυπα για τα συστήματα CAL.

4) Απαιτείται εξειδικευμένο προσωπικό και γενικότερη αποδοχή των συστημάτων CAL (συντήρηση συστημάτων, εκσυγχρονισμός διδασκόμενης ύλης κλπ)

5) Τα συστήματα CAL θεωρούνται από πολλούς υπεύθυνα για τη μηχανοποίηση της εκπαίδευσης, αποδυνάμωση των ανθρώπινων δεξιοτήτων (deskilling), και εξάρτηση από τον υπολογιστή.

6) Δεν υπάρχει κοινά αποδεκτή μέθοδος αξιολόγησης των συστημάτων CAL.

7) Ανεργία των εκπαιδευτικών

### 13.3 Ασφάλεια υπολογιστικών συστημάτων

Σήμερα οι υπολογιστές και η διασύνδεσή τους στο Διαδίκτυο και γενικώτερα σε δίκτυα υπολογιστών παίζουν ένα ρόλο όλο και πιο σημαντικό σε όλους τους τομείς, ταυτόχρονα όμως δημιουργούνται αυξανόμενα προβλήματα και προκλήσεις όσον αφορά την ασφάλεια των πληροφοριακών συστημάτων και των πληροφοριών που καταχωρούνται ή διακινούνται σε αυτά, απέναντι σε προσωρινή ή μόνιμη μη λειτουργία, φυσική καταστροφή, υποκλοπή πληροφοριών, κακόβουλες ενέργειες κ.α. Η ασφάλεια υπολογιστικών συστημάτων περιλαμβάνει συνήθως τα ακόλουθα :

α) Φυσική ασφάλεια. Προστασία από γεγονότα που μπορούν να προκαλέσουν μερική ή ολική καταστροφή του συστήματος (διακοπή παροχής ηλεκτρικού ρεύματος, υπερτάσεις, σεισμός, φωτιά, εκρήξεις κλπ).

β) Ασφάλεια πρόσβασης. Επιτρέπεται μόνο σε ορισμένους (εξουσιοδοτημένους) χρήστες να εργάζονται και να αναπτύσσουν εφαρμογές στο Σύστημα και να καθορίζουν πότε και από ποιόν θα χρησιμοποιούνται οι εφαρμογές αυτές.

γ) Προστασία των πληροφοριών κατά τη μετάδοσή τους μέσα από γραμμές επικοινωνίας και παρεμπόδιση παρεμβολών και υποκλοπών.

Στη φυσική ασφάλεια εκτός από συμβάντα που προκαλούν μερική ή ολική καταστροφή περιλαμβάνεται και προστασία από είσοδο μη εξουσιοδοτημένων ατόμων στο χώρο που βρίσκεται το σύστημα. Αυτή επιτυγχάνεται είτε με τη χρησιμοποίηση προσωπικού φύλαξης είτε με αυτόματα συστήματα ελέγχου πρόσβασης (access control). Τα συστήματα αυτά βασίζονται σε χρήση κωδικών /passwords μαγνητικών ή οπτικών καρτών κλπ. Πιο σύγχρονα συστήματα βασίζονται π.χ. στην αναγνώριση της φωνής του εισερχομένου, της υπογραφής, ή των δακτυλικών αποτυπωμάτων (βιομετρικά δεδομένα).

Για την αποτροπή της χρήσης εφαρμογών και υπηρεσιών του συστήματος από μη εξουσιοδοτημένους χρήστες χρησιμοποιείται στις περισσότερες περιπτώσεις η τεχνική των κωδικών /passwords. Για μεγαλύτερη ασφάλεια χρησιμοποιούνται :

- Πολλαπλά επίπεδα κωδικοποίησης
- Δυνατότητα καθορισμού υπηρεσιών και προγραμμάτων ώστε να έχουν πρόσβαση σε αυτά ορισμένοι μόνο χρήστες.

Για ασφάλεια κατά τη μετάδοση της πληροφορίας, χρησιμοποιούνται μέθοδοι που βασίζονται στην κρυπτογράφηση, δηλαδή την κωδικοποίηση της πληροφορίας, σύμφωνα με κάποιον κώδικα. Σύμφωνα με μια μέθοδο κρυπτογράφησης, χρησιμοποιείται ένας αριθμός χαρακτήρων που ονομάζεται κλειδί και προστίθεται στο αρχικό μήνυμα για να δώσει το κρυπτογραφημένο. Μόνο ο κάτοχος του κλειδιού (που μπορεί α αλλάζει) μπορεί να αποκρυπτογραφήσει το μήνυμα.

δ) Κακόβουλο λογισμικό (malware), ονομάζουμε προγράμματα που εισάγονται ή προσκολλούνται σε άλλα άλλα προγράμματα που είναι εγκατεστημένα σε κάποιον Η/Υ και προσπαθούν να μεταδοθούν μόνα τους και σε άλλους διασυνδεδεμένους Η/Υ. Μπορεί να θέλουν να υποκλέψουν δεδομένα/πληροφορίες (spyware), άλλα είναι απλώς ενοχλητικά και παρεμβατικά (μεταξύ άλλων προβάλλουν διαφημίσεις ή/και καθυστερούν τη λειτουργία του ΗΥ), άλλα όμως μπορεί να καταστρέψουν δεδομένα, όπως είναι οι ιοί (virus) και τα σκουλήκια (worms).

Phising (ψάρεμα) είναι η προσπάθεια από κακόβουλο λογισμικό απόσπασης προσωπικών ή εμπιστευτικών πληροφοριών από τον χρήστη.

Τα cookies χρησιμοποιούνται από εταιρίες για να παρέχουν στους διαδικτυακούς τόπους πληροφορίες σχετικά με τα ενδιαφέροντα και τις προτιμήσεις των χρηστών.

Χάκερ (hacker), είναι αυτός που εισβάλει παράνομα σε ένα υπολογιστικό σύστημα.

Για την προστασία των υπολογιστικών συστημάτων από κακόβουλο λογισμικό έχουν αναπτυχθεί (και διαρκώς αναπτύσσονται) ειδικά προγράμματα, γνωστά με τις ονομασίες firewall (τείχος προστασίας), antivirus κ,α. Παράλληλα χρησιμοποιούνται και άλλες μέθοδοι προστασίας, όπως η λήψη και τήρηση εφεδρικών αρχείων (backup files).

**ΒΙΒΛΙΟΓΡΑΦΙΑ**

Αρχιτεκτονική Υπολογιστών, Γ. Παπακωνσταντίνου Π.Τσανάκας Γ.Π. Φραγκάκης Εκδόσεις ΣΥΜΜΕΤΡΙΑ 1999

Εισαγωγή στην Πληροφορική, Alan Evans, Kendall Martin, Ann Poatsy , Μετάφραση : Αγαμέμνων Μήλιος, Επιμέλεια : Γιάννης Σταματίου , Εκδόσεις ΚΡΙΤΙΚΗ 3η έκδοση Ιούνιος 2022.

Πληροφοριακά Συστήματα Διοίκησης, Kenneth Laudon, Jane Laudon, Μετάφραση : Παναγιώτης Καναβός, Εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ, 2021

Πληροφορική - Συμπληρωματικό Εκπαιδευτικό Υλικό Γ' Τάξη Γενικού Λυκείου - Αναθ. Έκδοση 22-0279 ΙΤΥΕ ΔΙΟΦΑΝΤΟΣ