ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Λειτουργικά Συστήματα

Ενότητα # 1: Εισαγωγή

Διδάσκων: Γεώργιος Ξυλωμένος

Τμήμα: Πληροφορικής

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Οικονομικό Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

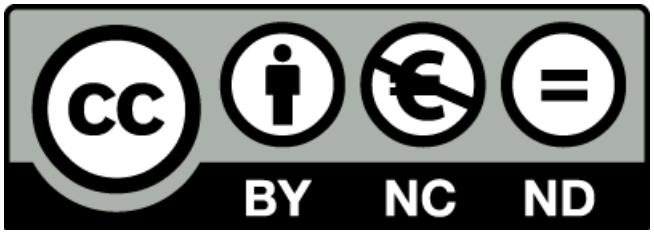
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Οι εικόνες προέρχονται από το βιβλίο «Σύγχρονα Λειτουργικά Συστήματα», A.S. Tanenbaum, 4η έκδοση, 2018, Εκδόσεις Κλειδάριθμος.



Σκοποί ενότητας

- Κατανόηση των στόχων και της ιστορικής εξέλιξης των λειτουργικών συστημάτων (ΛΣ)
- Επισκόπηση του υλικού υπολογιστών
- Κατανόηση των τύπων των ΛΣ
- Παρουσίαση των βασικών εννοιών και των τυπικών κλήσεων των ΛΣ
- Παρουσίαση των τρόπων δόμησης ΛΣ
- Εισαγωγή στη γλώσσα C

Περιεχόμενα ενότητας

- Τι είναι το λειτουργικό σύστημα (ΛΣ);
- Ιστορία ΛΣ
- Υλικό υπολογιστών
- Είδη ΛΣ
- Έννοιες ΛΣ
- Κλήσεις συστήματος
- Δομή ΛΣ
- Η γλώσσα C

Τι είναι το λειτουργικό σύστημα;

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Τι δεν είναι το ΛΣ;

- Δεν είναι το λογισμικό που έρχεται με τον ΗΥ
  - Μπορούμε να το αλλάξουμε αν θέλουμε
- Δεν είναι η διεπαφή με το χρήστη
  - Το ίδιο ΛΣ μπορεί να έχει GUI και CLI
- Δεν είναι ένα πρόγραμμα εφαρμογής
  - Δεν κάνει κάτι εμφανώς χρήσιμο για τον χρήστη

Τι κάνει το ΛΣ;

- Ένας υπολογιστής έχει πολλούς πόρους
  - Επεξεργαστές, μνήμες, δίσκους, εκτυπωτές, …
- Το ΛΣ είναι ένα ιδιαίτερο πρόγραμμα
  - Παροχή διεπαφής προς τον προγραμματιστή
    - Όχι προς τον απλό χρήστη!
  - Διαχείριση των πόρων του υπολογιστή
    - Χρονοπρογραμματισμός, έλεγχος πρόσβασης, …

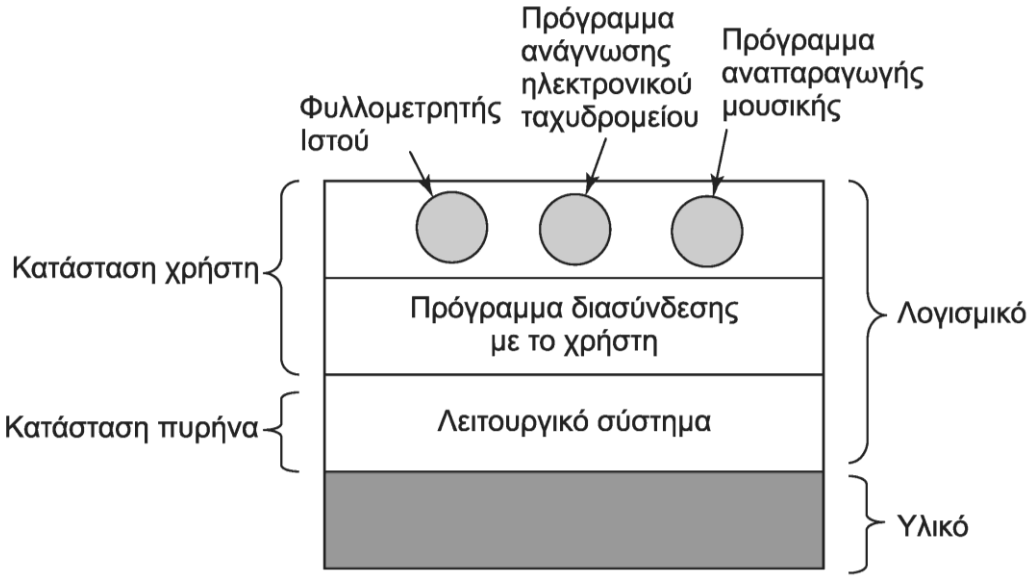
Η επεκτεταμένη μηχανή

- Όψη του λειτουργικού συστήματος από πάνω
  - Μετατροπή μιας άσχημης εικόνας σε πιο όμορφη
- Το λειτουργικό σύστημα παρέχει αφαιρέσεις
  - Κάθε είδος δίσκου είναι πολύ διαφορετικό
  - Το λειτουργικό σύστημα μας παρουσιάζει αρχεία
    - Μετατροπή εντολών αρχείων σε εντολές δίσκου
- Σε ποιον παρέχεται η λογική αφαίρεση;
  - Στους προγραμματιστές των εφαρμογών
    - Διεπαφή προγραμματισμού (API)

Ο διαχειριστής πόρων

- Όψη του λειτουργικού συστήματος από κάτω
  - Διαχείριση των πόρων ενός συστήματος
- Ταυτόχρονη εκτέλεση προγραμμάτων
  - Κάθε πρόγραμμα θεωρεί ότι έχει δική του μηχανή
- Διαχείριση και προστασία πόρων
- Πολύπλεξη πόρων σε δύο άξονες
  - Χρόνος: διαδοχική χρήση ΚΜΕ από προγράμματα
  - Χώρος: συνύπαρξη προγραμμάτων στη μνήμη

Καταστάσεις επεξεργαστή (1 από 2)



- Κατάσταση πυρήνα: λειτουργικό σύστημα
  - Επιτρέπεται η εκτέλεση όλων των εντολών
- Κατάσταση χρήστη: διεπαφή και εφαρμογές
  - Δεν επιτρέπεται η εκτέλεση ορισμένων εντολών

Καταστάσεις επεξεργαστή (2 από 2)

- Γιατί χρειάζονται οι καταστάσεις;
- Η πολύπλεξη πόρων απαιτεί προστασία
  - Τα προγράμματα πρέπει να απομονώνονται
  - Οι «επικίνδυνες» λειτουργίες είναι για το ΛΣ
- Το ΛΣ μας δίνει μια εικονική μηχανή
  - Επιτρέπονται όλες οι μη προνομιούχες εντολές
  - Οι προνομιούχες κρύβονται πίσω από κλήσεις ΛΣ

Γιατί να έχουμε μάθημα ΛΣ;

- Τι περιλαμβάνει το λειτουργικό σύστημα;
  - Λογισμικό που εκτελείται σε κατάσταση πυρήνα
  - Πιθανόν και προνομιούχο λογισμικό χρήστη
- Γιατί μελετάμε τα λειτουργικά συστήματα;
  - Κρίσιμος παράγοντας απόδοσης των εφαρμογών
  - Δύσκολο να γραφτούν νέα λόγω πολυπλοκότητας
- Κύρια λειτουργικά συστήματα
  - UNIX και απόγονοι: Linux, BSD, OSX
  - Windows NT και απόγονοι

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Ιστορία ΛΣ

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Γιατί μας ενδιαφέρει η ιστορία;

- Τα ΛΣ δεν σχεδιάστηκαν με κάποιο σχέδιο
  - Αρχικά δεν υπήρχαν καθόλου ΛΣ
  - Σταδιακά άρχισε να γράφεται τέτοιο λογισμικό
    - Κάθε φορά, αντιμετώπιζε συγκεκριμένη ανάγκη
    - Είτε εξελίξεις υλικού, είτε άλλους τρόπους χρήσης
- Τα σημερινά ΛΣ έχουν μακριά ιστορία
  - Η σχεδίασή τους αντανακλά ιστορικές επιλογές

Ιστορία ΛΣ (1 από 12)

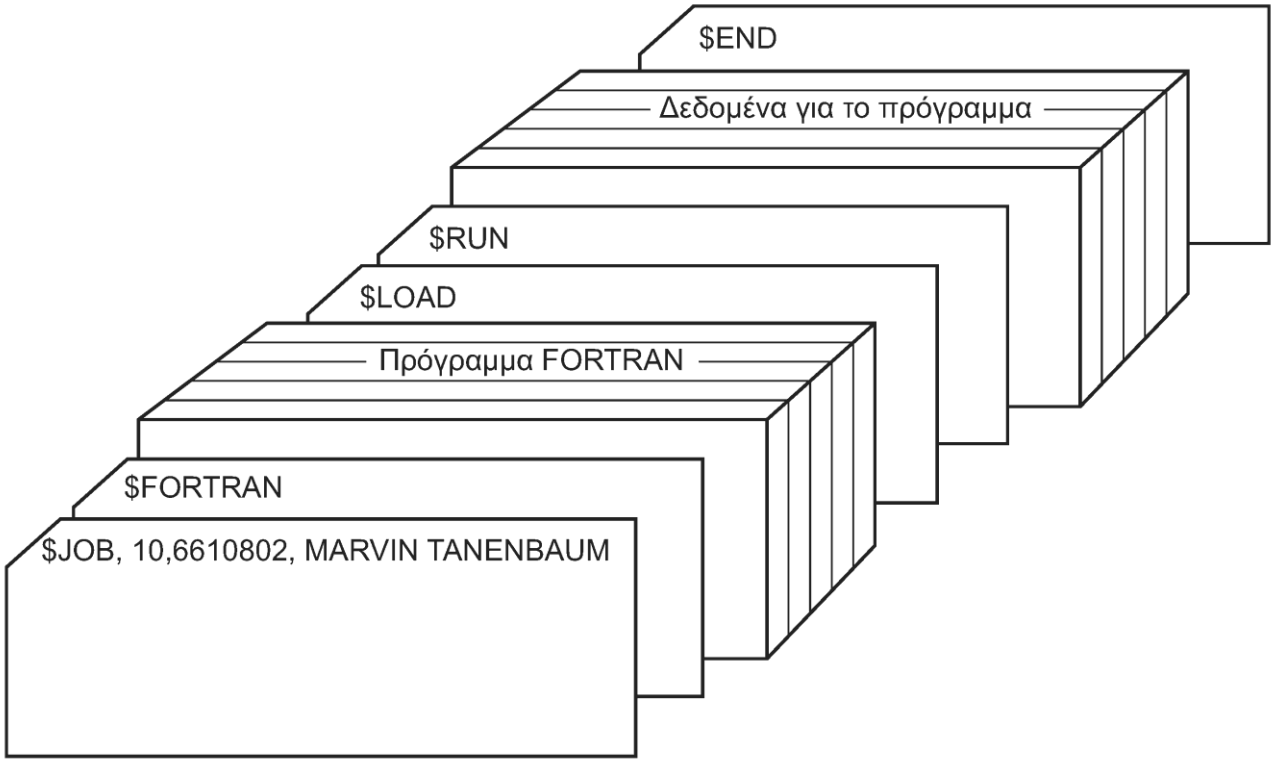
- Πρώτη γενιά (1945-1955): λυχνίες κενού
  - Προγραμματισμός από τους σχεδιαστές
  - Απόλυτη γλώσσα μηχανής ή πίνακες καλωδιώσεων
  - Σχετικά απλές μηχανές με απλές λειτουργίες
  - Δεν υπήρχε λειτουργικό σύστημα
- Χρήση διάτρητων καρτών για προγραμματισμό
  - Αρχικά χρήση μόνο για είσοδο δεδομένων
  - Αργότερα, αποθήκευση προγράμματος στη μνήμη

Ιστορία ΛΣ (2 από 12)

- Δεύτερη γενιά (1955-1965): τρανζίστορ
  - Μηχανές αρκετά αξιόπιστες ώστε να πωλούνται
  - Οι προγραμματιστές δεν είναι πια οι σχεδιαστές
- Υποβολή και εκτέλεση εργασιών (jobs)
  - Κάθε εργασία αποτελείται από μια σειρά κάρτες
  - Ο προγραμματιστής δίνει τις κάρτες στο χειριστή
  - Ο χειριστής υποβάλει τις εργασίες με τη σειρά
  - Χρειάζεται ένας ελεγκτής των εργασιών!

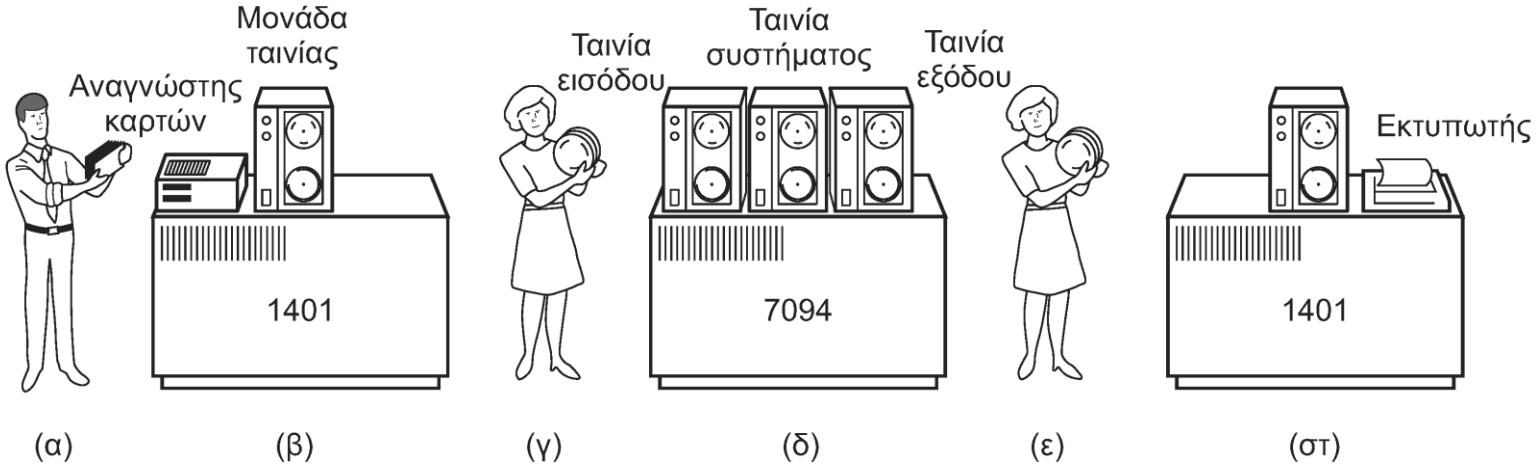
Ιστορία ΛΣ (3 από 12)

- Συστήματα δέσμης
  - Ειδικές κάρτες ελέγχου μιας εργασίας ($)
  - Κάρτες προγράμματος και δεδομένων



Ιστορία ΛΣ (4 από 12)

- Αυτοματοποίηση συστημάτων δέσμης
  - Αντιγραφή εργασιών σε ταινίες σε μικρό υπολογιστή
  - Υποβολή ταινίας με εργασίες σε μεγάλο υπολογιστή
  - Έξοδος αποτελεσμάτων σε άλλη ταινία
  - Εκτύπωση αποτελεσμάτων σε μικρό υπολογιστή



Ιστορία ΛΣ (5 από 12)

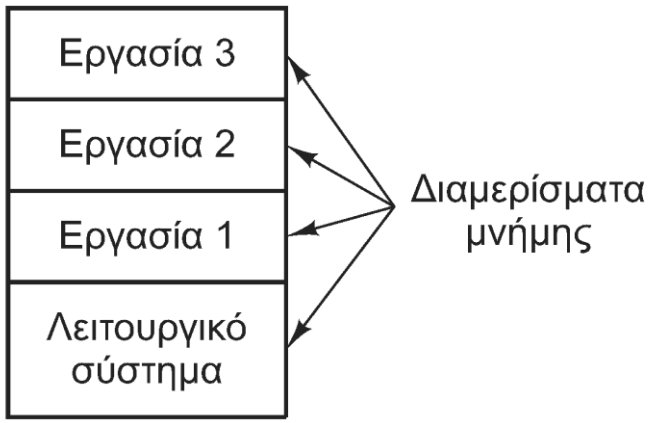
- Πρώτες γλώσσες προγραμματισμού
  - FORTRAN (1956), LISP (1958), COBOL (1959)
  - Μέχρι τότε, προγραμματισμός σε assembly!
  - Οι γλώσσες παρέχουν ένα επίπεδο αφαίρεσης
    - Κρύβουν τις συσκευές πίσω από αρχεία
  - Σταδιακά, μέρος της αφαίρεσης γίνεται το ΛΣ
    - Δεν χρειάζεται να υλοποιείται για κάθε γλώσσα
    - Οι γλώσσες προσθέτουν τις δικές τους δυνατότητες

Ιστορία ΛΣ (6 από 12)

- Τρίτη γενιά (1965-1980): ολοκληρωμένα κυκλώματα
  - Αντικατάσταση τρανζίστορ με κυκλώματα SSI
  - Προσανατολισμός σε συγκεκριμένες εφαρμογές
- Οικογένεια υπολογιστών
  - Ιδέα της IBM με το System/360
  - Μηχανές με ίδια αρχιτεκτονική και σύνολο εντολών
  - Διαφορετικές δυνατότητες και κόστος
  - Ίδια προγράμματα (εντός λογικών πλαισίων)
  - Χρειάζεται ένα ΛΣ να κρύβει τις διαφορές

Ιστορία ΛΣ (7 από 12)

- Πολυπρογραμματισμός
  - Διακοπτόμενη εκτέλεση πολλών εργασιών
  - Κάθε εργασία βρίσκεται σε χωριστό μέρος τη μνήμης
  - Όταν μια εργασία περιμένει, εκτελείται κάποια άλλη
  - Ταυτόχρονη επεξεργασία και Ε/Ε (spooling)
  - Στόχος: Καλύτερη αξιοποίηση πόρων



Ιστορία ΛΣ (8 από 12)

- Χρονομερισμός
  - Ένα βήμα πέρα από τον πολυπρογραμματισμό
  - Κάθε χρήστης έχει το δικό του τερματικό
  - Το σύστημα εξυπηρετεί χρήστες ψευδοταυτόχρονα
  - Στόχος: καλύτερη εξυπηρέτηση χρηστών
- Το σύστημα MULTICS
  - Υπολογιστική ισχύς από κεντρικό υπολογιστή
- Τα συστήματα DEC PDP
  - Μικροί και (σχετικά οικονομικοί) υπολογιστές

Ιστορία ΛΣ (9 από 12)

- MULTICS+PDP = UNIX
  - Επηρέασε όλα τα μεταγενέστερα συστήματα
  - Το UNIX διακλαδώθηκε σε System V και BSD
  - Το MINIX γράφτηκε για εκπαιδευτικούς σκοπούς
    - Μικροπυρήνας και έμφαση στην απλότητα
    - Στόχος: UNIX που να το καταλαβαίνουν οι φοιτητές
  - Το Linux γράφτηκε ως πρακτικό MINIX
    - Μονολιθικός πυρήνας και έμφαση στη λειτουργικότητα
    - Στόχος: ένα UNIX που να κάνει οποιαδήποτε δουλειά

Ιστορία ΛΣ (10 από 12)

- Τέταρτη γενιά (1980-): προσωπικοί υπολογιστές
  - Χρήση κυκλωμάτων LSI και αργότερα VLSI
  - Δυνατότητα τοποθέτησης ΚΜΕ σε ένα ολοκληρωμένο
  - Εμφάνιση προσωπικών υπολογιστών
- ΛΣ προσωπικών υπολογιστών
  - Το CP/M γράφτηκε για 8080 (αργότερα, για Z80)
  - Το MS-DOS γράφτηκε για 8088/8086
    - Αρχικά σαν το CP/M, μετά με ιδέες από το UNIX
  - Εμφάνιση υπολογιστών με γραφικές διεπαφές

Ιστορία ΛΣ (11 από 12)

- Τα παλιά Microsoft Windows
  - Γραφική διεπαφή πάνω από το MS-DOS
  - Windows 3, 95, 98, Me
- Τα νέα Microsoft Windows
  - Νέα υλοποίηση ξεκινώντας από τα Windows NT
  - Windows 2000, XP, 7, Vista, 8, 10
- Τα συστήματα UNIX
  - Προσθήκη γραφικής διεπαφής: X Window System
  - Πολύ μεγάλη εξάπλωση του Linux
  - Το UNIX (τύπου BSD) αντικατέστησε και το Mac OS

Ιστορία ΛΣ (12 από 12)

- Πέμπτη γενιά (1990-): έξυπνες συσκευές
  - PDA, τηλέφωνα, ταμπλέτες
- Symbian: Συνηθισμένο στα πρώτα κινητά
- iOS: Παρουσιάστηκε με το iPhone
  - Προγραμματισμός σε Objective C
- Android: Εναλλακτική λύση της Google
  - Βασίζεται στο Linux
  - Προγραμματισμός σε Java

Υλικό υπολογιστών

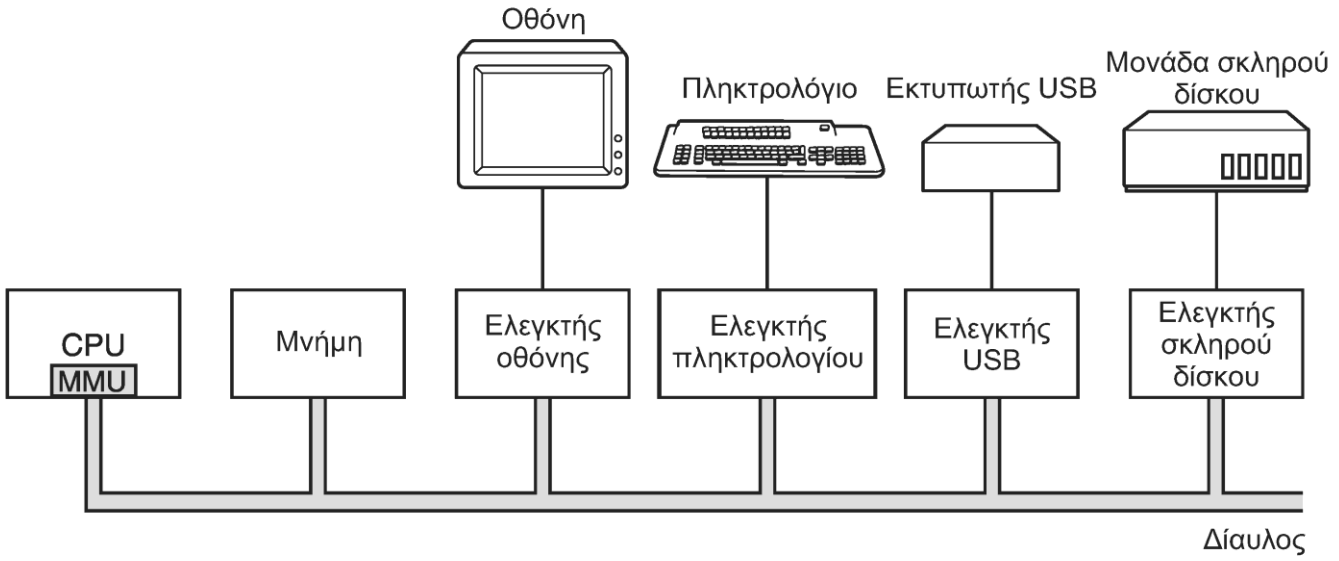
Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Υλικό δεν κάνει η ΟΣΥ;

- Ναι, οπότε θα είμαστε σύντομοι
- Αλλά, στα ΛΣ το υλικό έχει σημασία
  - Το ΛΣ πολυπλέκει τους πόρους
  - Το ΛΣ κρύβει το υλικό κάτω από αφαιρέσεις
- Μας ενδιαφέρουν δύο κυρίως σημεία
  - Η διεπαφή με το λογισμικό
  - Η αξιοποίηση των πόρων

ΛΣ και υλικό

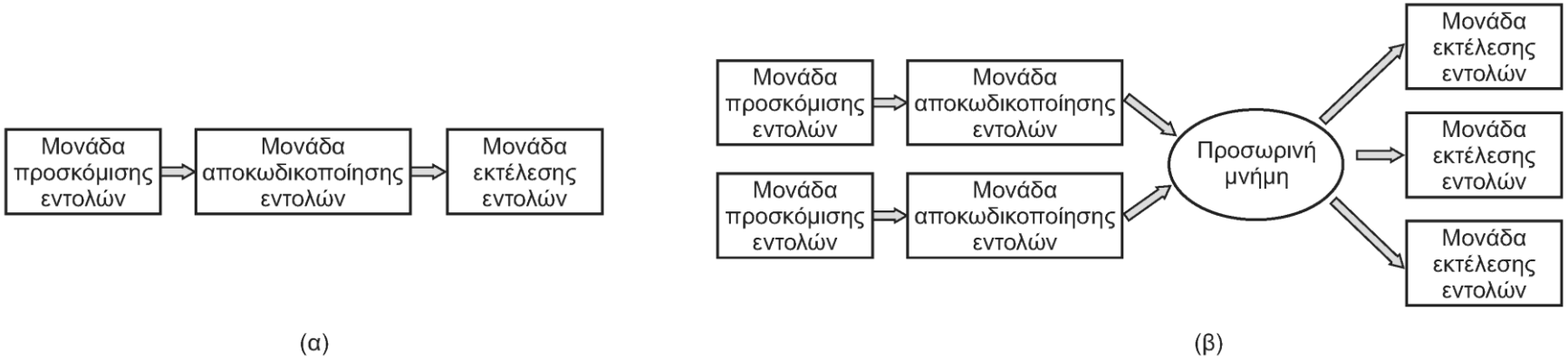


- Λειτουργικό σύστημα και υλικό
  - Βασική δομή υπολογιστικού συστήματος
  - Σύνδεση των στοιχείων μέσω διαύλου
    - Μπορεί να υπάρχουν πολλοί δίαυλοι

Επεξεργαστής (1 από 4)

- Επεξεργαστής
  - Βασικός κύκλος λειτουργίας
    - Προσκόμιση, αποκωδικοποίηση, τελεστές, εκτέλεση
  - Καταχωρητές δεδομένων
  - Καταχωρητές ελέγχου
    - Μετρητής προγράμματος, δείκτης στοίβας, κατάσταση
  - Αποθήκευση καταχωρητών κατά την εναλλαγή
    - Λειτουργία με αρκετά μεγάλο κόστος

Επεξεργαστής (2 από 4)

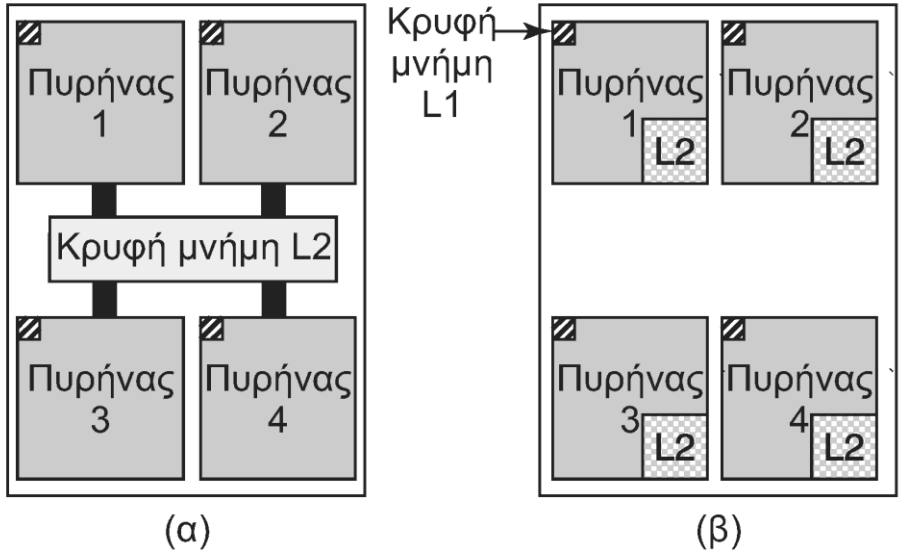


- Σωλήνωση
  - Χωριστές μονάδες για κάθε στάδιο εκτέλεσης
  - Εκτέλεση διαφορετικής εντολής σε κάθε ομάδα
- Υπερβαθμωτοί επεξεργαστές
  - Πολλαπλές μονάδες εκτέλεσης
  - Προσκόμιση και αποκωδικοποίηση πολλών εντολών
  - Εκτέλεση εντολών και εκτός σειράς

Επεξεργαστής (3 από 4)

- Καταστάσεις λειτουργίας επεξεργαστή
  - Κατάσταση χρήστη και κατάσταση πυρήνα
  - Σε κατάσταση χρήστη μόνο υποσύνολο εντολών
- Κλήσεις συστήματος
  - Εντολές που προκαλούν παγίδα υλικού
  - Ελεγχόμενη μετάβαση σε κατάσταση πυρήνα
- Πολυνηματικοί και πολυπύρηνοι επεξεργαστές
  - Πολλοί τρόποι αξιοποίησης διαθέσιμων τρανζίστορ
  - Κρυφή μνήμη, πολλαπλές μονάδες εκτέλεσης
  - Γιατί όχι και πολλές μονάδες ελέγχου;

Επεξεργαστής (4 από 4)



- Πολυνηματικοί και πολυπύρηνοι επεξεργαστές
  - Πολυνημάτωση: εκτέλεση πολλών νημάτων
  - Πολυπύρηνος επεξεργαστής: ανεξάρτητες CPU
    - Διάφορες σχεδιάσεις κρυφής μνήμης

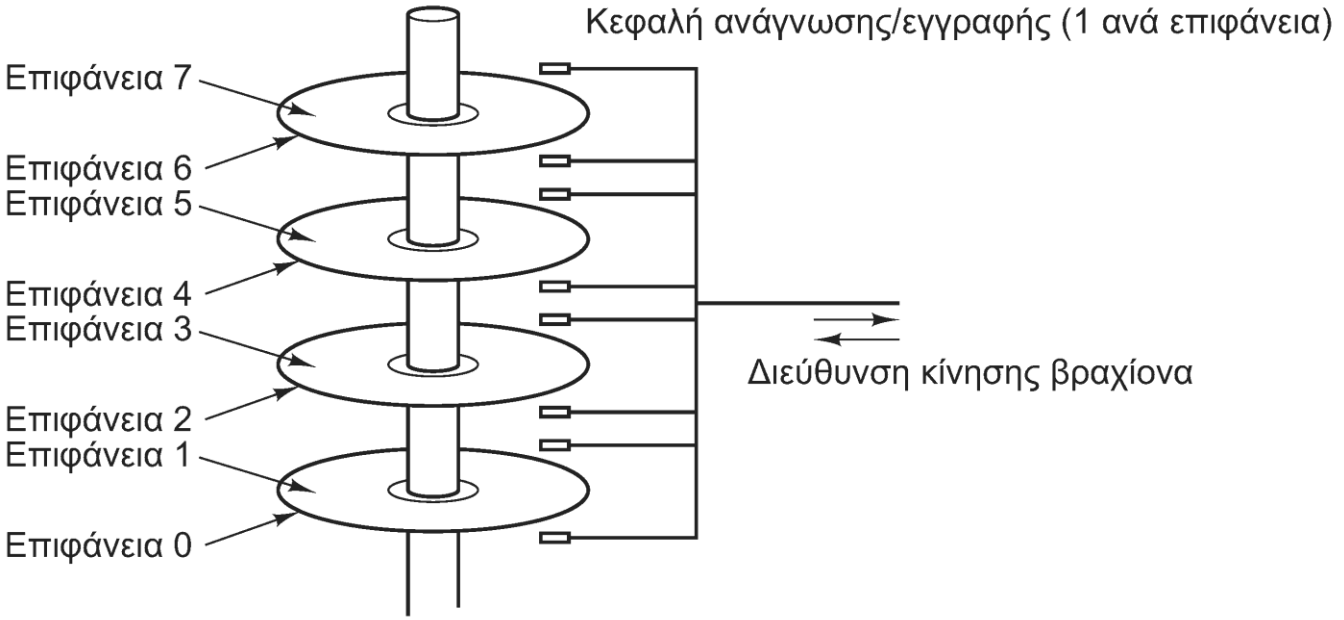
Μνήμη (1 από 2)

- Μνήμη και ιεραρχία μνήμης
  - Κάθε τύπος μνήμης παρέχει έναν συμβιβασμό
  - Ταχύτητα έναντι κόστους (άρα και μεγέθους)
  - Μνήμη ROM ή Flash για βασικό λογισμικό
  - Μνήμη CMOS για διάρθρωση συστήματος

Μνήμη (2 από 2)

- Κρυφές μνήμες
  - Οργανώνονται και αυτές ιεραρχικά
  - Γενικά θέματα οργάνωσης κρυφών μνημών
  - Πολιτικές προσκόμισης, τοποθέτησης, αντικατάστασης
- Κρυφή μνήμη επιπέδου 1 (L1)
  - Στο εσωτερικό του επεξεργαστή
  - Συνήθως χωριστή για εντολές και δεδομένα
- Κρυφή μνήμη επιπέδου 2 (L2)
  - Εντός ή εκτός επεξεργαστή
  - Κοινή ή καταμεριζόμενη ανάμεσα στους πυρήνες

Αποθήκευση

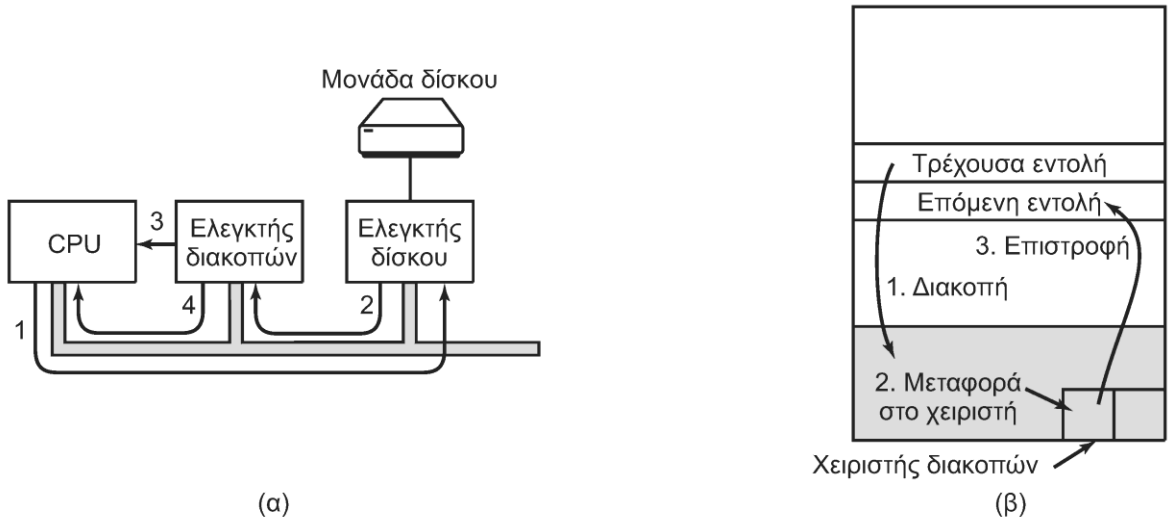


- Μαγνητικοί δίσκοι
  - Αρκετός χρόνος μέχρι να βρεθεί το σωστό σημείο
- Δίσκοι SSD (μνήμη flash)
  - Αμελητέα αναζήτηση, πιο αργή εγγραφή από ανάγνωση

Είσοδος / Έξοδος (1 από 2)

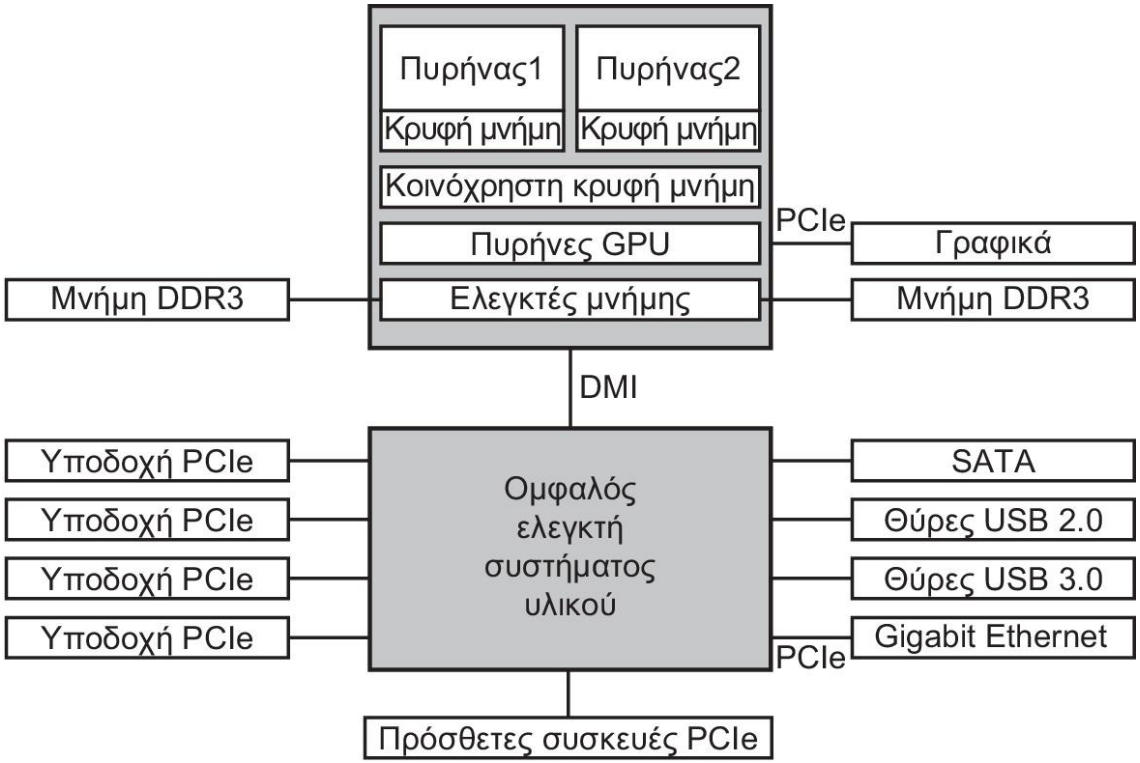
- Συσκευές εισόδου/εξόδου
  - Ελεγκτής και πραγματική συσκευή
  - Ο ελεγκτής παρουσιάζει απλούστερη διεπαφή
  - Οδηγός συσκευής: λογισμικό για τον ελεγκτή
  - Επικοινωνία με τους ελεγκτές των συσκευών
    - Απεικόνιση καταχωρητών στη μνήμη
    - Χωριστές εντολές εισόδου/εξόδου
  - Έλεγχος συσκευών: polling, interrupts, DMA

Είσοδος / Έξοδος (2 από 2)



- Είσοδος/έξοδος με διακοπές
  - Ο ελεγκτής διακόπτει την CPU όταν ολοκληρώσει
  - Μεταφορά ελέγχου σε χειριστή διακοπής
- Είσοδος/έξοδος με DMA
  - Ο ελεγκτής μεταφέρει ο ίδιος τα δεδομένα στη μνήμη

Δίαυλοι (1 από 2)



- Λεωφόροι (δίαυλοι) συστήματος
  - Γρήγορες και αργές συσκευές

Δίαυλοι (2 από 2)

- Δίαυλοι συστήματος
  - Άμεση επικοινωνία με μνήμη και γραφικά
  - Δίαυλος DMI για επικοινωνία με άλλες συσκευές
    - Χωριστό κύκλωμα για διασύνδεση συσκευών
  - Δίαυλος PCI Express για τις περισσότερες συσκευές
  - Δίαυλοι USB για σειριακές συσκευές
  - Δίαυλοι SCSI και SATA για δίσκους
  - Τοποθέτηση και άμεση λειτουργία (plug and play)

Εκκίνηση ΛΣ

- Βασικό λογισμικό συστήματος στο BIOS
  - Αναζητεί βασικές συσκευές σε γνωστές θέσεις
  - Εξετάζει διαύλους για πρόσθετες συσκευές
  - Καθορισμός συσκευής εκκίνησης μέσω CMOS
  - Φόρτωση τομέα 0 από συσκευή εκκίνησης
  - Προσδιορισμός ενεργής διαμέρισης
  - Φόρτωση προγράμματος εκκίνησης από τη διαμέριση
  - Φόρτωση λειτουργικού συστήματος
  - Εξέταση συσκευών από BIOS και φόρτωση οδηγών
  - Αρχικοποίηση δομών δεδομένων και φόρτωση φλοιού

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Είδη ΛΣ

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Είδη ΛΣ (1 από 3)

- ΛΣ για μεγάλους υπολογιστές (mainframes)
  - Πολύ μεγάλες ικανότητες εισόδου/εξόδου
  - Επεξεργασία δέσμης, συναλλαγών, χρονομερισμός
- ΛΣ για διακομιστές (servers)
  - Διαχείριση αιτημάτων πάρα πολλών πελατών
  - Μεγάλοι υπολογιστές ή ισχυροί μικροϋπολογιστές
- ΛΣ για πολυεπεξεργαστές (multiprocessors)
  - Από παράλληλοι ως πολυπύρηνοι υπολογιστές
  - Τώρα πια όλα τα συστήματα είναι τέτοια!

Είδη ΛΣ (2 από 3)

- ΛΣ για υπολογιστές χειρός (handhelds)
  - Smartphones και tablets
  - Χωρίς δίσκους, αποθήκευση σε flash
- Ενσωματωμένα ΛΣ (embedded)
  - DVD, τηλέφωνα (όχι smart), media players
  - Εκτελούν μόνο προκαθορισμένα προγράμματα
- ΛΣ κόμβων αισθητήρων
  - Ακόμη πιο απλά από τα ενσωματωμένα ΛΣ
  - Έμφαση στην εξοικονόμηση ενέργειας

Είδη ΛΣ (3 από 3)

- ΛΣ πραγματικού χρόνου
  - Αυστηρά συστήματα πραγματικού χρόνου
    - Όλες οι εργασίες πρέπει να εκτελούνται εγκαίρως
    - Συστήματα ελέγχου, ηλεκτρονικά αεροσκαφών
  - Ήπια συστήματα πραγματικού χρόνου
    - Οι εργασίες πρέπει να εκτελούνται συνήθως εγκαίρως
    - Κινητά τηλέφωνα, media players
- ΛΣ έξυπνων καρτών
  - Παράδειγμα: κάρτες για ψηφιακές υπογραφές

Έννοιες ΛΣ

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

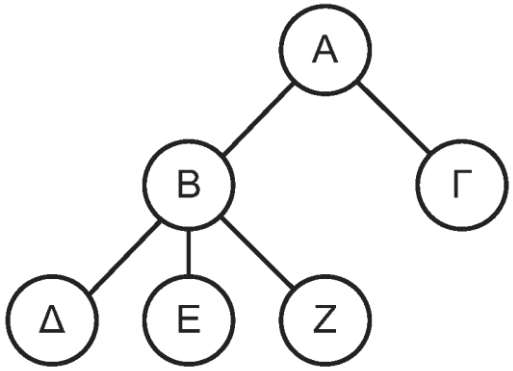
Επισκόπηση του ΛΣ

- Στα ΛΣ έχουμε μια σειρά από νέες έννοιες
  - Διεργασίες: εκτελούμενα προγράμματα
  - Χώροι διευθύνσεων: μνήμη διεργασιών
  - Αρχεία: όχι και τόσο νέα έννοια...
- Οι έννοιες αυτές είναι οι λογικές αφαιρέσεις
- Στόχος: να δούμε την μεγάλη εικόνα
- Κάθε έννοια αναλύεται πλήρως παρακάτω

Διεργασίες (1 από 2)

- Διεργασίες: προγράμματα που εκτελούνται
  - Χώρος διευθύνσεων: πρόγραμμα και δεδομένα
  - Πόροι: καταχωρητές, ανοιχτά αρχεία, σήματα
- Προσωρινή αναστολή διεργασίας
  - Αποθήκευση πόρων στον πίνακα διεργασιών
  - Ο πίνακας διεργασιών είναι συνέχεια στη μνήμη
  - Χώρος διευθύνσεων (πιθανόν) εν μέρει στη μνήμη
- Κλήσεις διαχείρισης διεργασιών
  - Δημιουργία και τερματισμός διεργασιών
  - Επικοινωνία διεργασιών (όταν αυτό επιτρέπεται)

Διεργασίες (2 από 2)



- Δένδρα διεργασιών
- Σήματα (signals)
  - Ειδοποιήσεις προς τη διεργασία για κάποιο γεγονός
  - Παράδειγμα: εκπνοή χρονομέτρου (alarm)
- Προνόμια διεργασίας
  - Κωδικός ταυτότητας χρήστη (UID)
  - Κωδικός ταυτότητας ομάδας (GID)

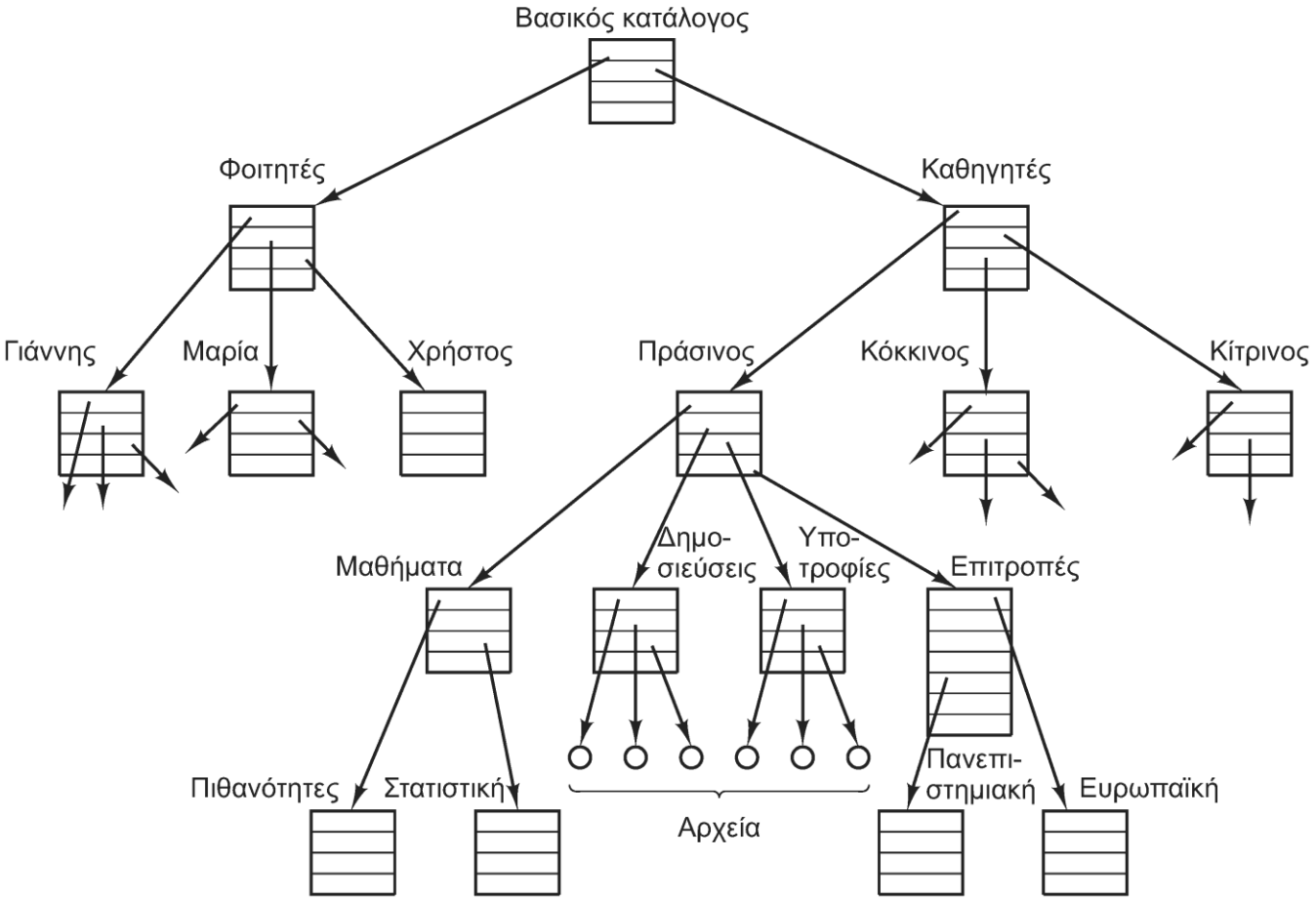
Χώροι διευθύνσεων

- Χώροι διευθύνσεων
  - Τα ΛΣ γενικά υποστηρίζουν πολλές διεργασίες
  - Μηχανισμός προστασίας κάθε διεργασίας
    - Απομόνωση της μνήμης της
  - Εικονική μνήμη: μεγάλος χώρος διευθύνσεων
    - Δεν είναι όλος ταυτόχρονα στην κύρια μνήμη
    - Διευκολύνει τη συνύπαρξη διεργασιών στη μνήμη
    - Ό,τι δεν είναι στην κύρια μνήμη είναι στο δίσκο

Αρχεία (1 από 4)

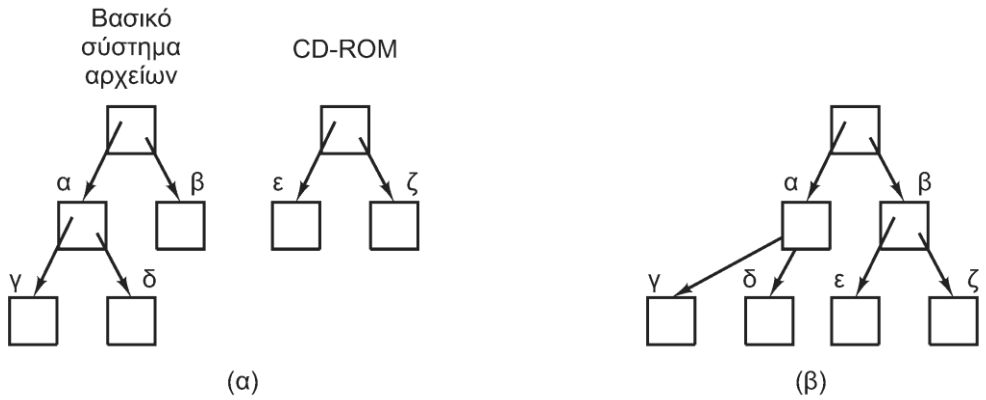
- Αρχεία
  - Λογική όψη συσκευών αποθήκευσης
    - Ενιαία μορφή αρχείων σε όλες τις συσκευές
  - Οργάνωση αρχείων σε καταλόγους
  - Περιγραφέας αρχείου: δείκτης σε ανοιχτό αρχείο
    - Χρησιμοποιείται από τις διεργασίες
    - Χειριστήριο για πράξεις στα αρχεία
  - Κατάλογος εργασίας διεργασίας

Αρχεία (2 από 4)



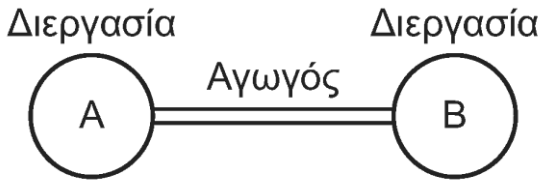
- Παράδειγμα δενδρικής οργάνωσης αρχείων

Αρχεία (3 από 4)



- Όνομα διαδρομής αρχείου
  - Απόλυτο: ξεκινάει από τη ρίζα
  - Σχετικό: ξεκινάει από τον κατάλογο εργασίας
- Ανάρτηση συστημάτων αρχείων
  - Εκκίνηση με το βασικό σύστημα αρχείων
  - Ανάρτηση πρόσθετων συστημάτων ανάλογα με τις ανάγκες

Αρχεία (4 από 4)



- Ειδικά αρχεία
  - Ειδικά αρχεία μπλοκ ή ομάδων
    - Συσκευές με τυχαία προσπελάσιμα μπλοκ (π.χ. δίσκοι)
  - Ειδικά αρχεία χαρακτήρων
    - Συσκευές με ρεύματα χαρακτήρων (π.χ. πληκτρολόγια)
- Αγωγοί ή σωληνώσεις (pipes)
  - Ψευδοαρχείο ανάμεσα σε δύο διεργασίες
  - Επιτρέπει την επικοινωνία των διεργασιών

Άλλες έννοιες

- Είσοδος/έξοδος
  - Υποσύστημα διαχείρισης συσκευών εισόδου/εξόδου
  - Οδηγοί συσκευών ανά κατηγορία συσκευής
- Προστασία
  - Μηχανισμός που προστατεύει στοιχεία διεργασιών
  - Παράδειγμα: bit προστασίας στο UNIX
- Κέλυφος
  - Δεν είναι μέρος του λειτουργικού (αλλάζει «εύκολα»)
  - Επιτρέπει στον χρήστη να επικοινωνεί με το λειτουργικό
  - Γραμμή εντολών ή γραφική διεπαφή

Ανακύκλωση ιδεών (1 από 3)

- Υπολογιστές και ΛΣ ανακυκλώνουν ιδέες
  - Οι αλλαγές στην τεχνολογία οδηγούν σε νέες ιδέες
  - Οι παλιές επανέρχονται μετά από άλλες αλλαγές
  - Δεν υπάρχουν απαρχαιωμένες, μόνο μη επίκαιρες!
- Υλοποίηση CPU
  - Αρχικά καλωδιωμένη (πιο απλή)
  - Στη συνέχεια μικροπρογραμματιζόμενη (πιο ευέλικτη)
  - Στα RISC, πάλι καλωδιωμένη (πιο αποδοτική)

Ανακύκλωση ιδεών (2 από 3)

- Μεγάλες μνήμες
  - Οι υπολογιστές ξεκίνησαν με πολύ μικρές μνήμες
    - Όλα τα προγράμματα σε συμβολική γλώσσα
  - Η αύξηση μνήμης οδήγησε στους μεταγλωττιστές
  - Οι μίνι/μίκρο γύρισαν σε συμβολική γλώσσα
- Υλικό προστασίας
  - Οι μεγάλοι υπολογιστές δεν είχαν υλικό προστασίας
  - Το υλικό προστασίας επέτρεψε πολυπρογραμματισμό
  - Το ίδιο έγινε με μίνι/μίκρο-υπολογιστές

Ανακύκλωση ιδεών (3 από 3)

- Συσκευές αποθήκευσης
  - Αρχικά ενιαίος κατάλογος για όλο το σύστημα
  - Στη συνέχεια ένας κατάλογος ανά χρήστη
  - Τελικά δενδρικές και άλλες δομές
  - Ίδια ακριβώς εξέλιξη σε μίνι/μίκρο-υπολογιστές
- Εικονική μνήμη
  - Έγινε δυνατή με υλικό απεικόνισης και προστασίας
  - Αρχικά μόνο σε μεγάλους υπολογιστές
  - Στη συνέχεια σε μίνι/μίκρο-υπολογιστές

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Κλήσεις συστήματος

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

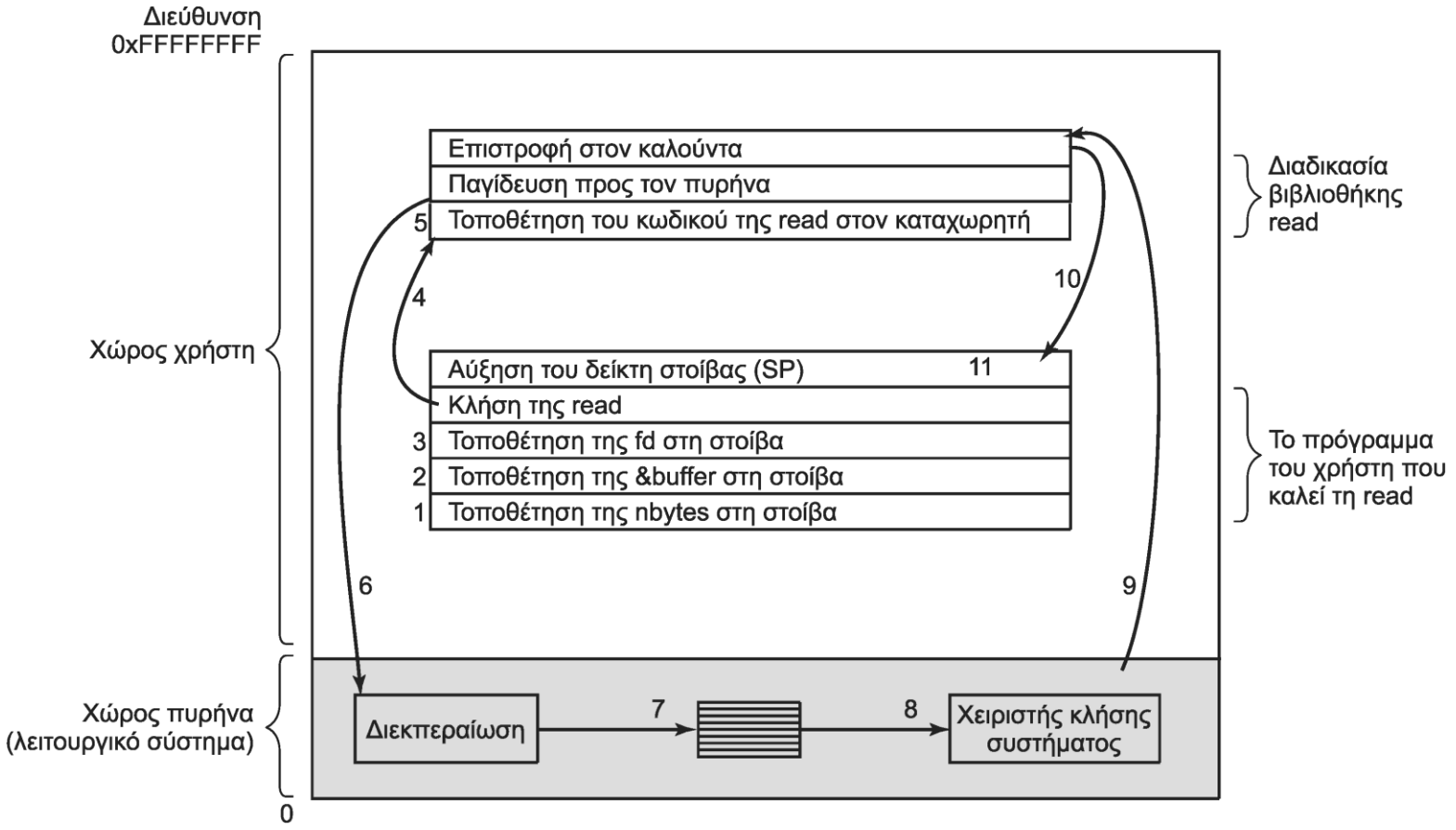
Τι κάνουν οι κλήσεις συστήματος;

- Οι κλήσεις συστήματος είναι είσοδοι στο ΛΣ
  - Μας επιτρέπουν να ζητάμε υπηρεσίες
  - Οδηγούν σε κατάσταση πυρήνα
    - Και εκτελούν εκεί κώδικα
  - Στο τέλος γυρνάμε σε κατάσταση χρήστη
- Οι κλήσεις αυτές είναι η διεπαφή του ΛΣ
  - Καθορίζουν τι μπορούμε να ζητήσουμε
  - Άρα, μας δείχνουν τι κάνει το σύστημα για εμάς

Κλήσεις συστήματος (1 από 5)

- Διασύνδεση προγραμμάτων με το ΛΣ
  - Η λογική αφαίρεση που παρουσιάζει το ΛΣ
  - Χρήση του POSIX (UNIX) ως παράδειγμα
  - Περιγράφεται με σειρά κλήσεων βιβλιοθήκης
    - Τα δύο είδη κλήσεων δεν έχουν απόλυτη αντιστοιχία
  - Οι κλήσεις ενθυλακώνουν τις κλήσεις συστήματος
    - Έλεγχος, μετάβαση σε κατάσταση συστήματος
    - Εκτέλεση εργασίας, επιστροφή σε κατάσταση χρήστη

Κλήσεις συστήματος (2 από 5)



- Παράδειγμα: count=read(fd,&buffer,nbytes);

Κλήσεις συστήματος (3 από 5)

- Βήματα 1-3: τοποθέτηση παραμέτρων στη στοίβα
- Βήμα 4: κλήση διαδικασίας read
- Βήμα 5: προετοιμασία παραμέτρων κλήσης
- Βήμα 6: χρήση εντολής TRAP για την κλήση
- Βήμα 7: μετάβαση σε χειριστή κλήσης
- Βήμα 8: εκτέλεση ζητούμενης εργασίας
- Βήμα 9: επιστροφή στην εντολή μετά την TRAP
- Βήμα 10: επιστροφή στον καλούντα
- Βήμα 11: καθάρισμα στοίβας

Κλήσεις συστήματος (4 από 5)

- Πού βρίσκονται οι κλήσεις συστήματος;
  - Σε κάποια βιβλιοθήκη συστήματος
    - Glibc στο Linux
  - Περιλαμβάνουν και γλώσσα μηχανής
    - Εντολή TRAP για παγίδευση
  - Καλούνται άμεσα ή έμμεσα
    - Άμεσα: κλήση της read από τη C
    - Έμμεσα: κλήση της fread ή της scanf από τη C

Κλήσεις συστήματος (5 από 5)

- Υλοποίηση κλήσης συστήματος
  - Εντολή TRAP n ή INT n ή SYSCALL n
    - Εναλλακτικά, το n είναι σε καταχωρητή
  - Αλλαγή κατάστασης λειτουργίας
  - Αποθήκευση μετρητή προγράμματος
  - Φόρτωση νέου μετρητή από πίνακα
    - Δημιουργείται από το ΛΣ κατά την εκκίνηση
    - Δείχνει στον κώδικα κάθε κλήσης

Διαχείριση διεργασιών (1 από 3)

Κλήση	Περιγραφή δράσης
pid = fork()	Δημιουργεί μια θυγατρική διεργασία, πανομοιότυπη με τη γονική
pid = waitpid(pid, &statloc, επιλογές)	Οδηγεί σε αναμονή μέχρι τον τερματισμό της θυγατρικής διεργασίας
s = execve(όνομα, argv, environp)	Αντικαθιστά την εικόνα πυρήνα μιας διεργασίας
exit(κατάσταση)	Τερματίζει την εκτέλεση της διεργασίας και επιστρέφει τον κωδικό κατάστασης

- Κλήσεις διαχείρισης διεργασιών
  - fork(): δημιουργεί αντίγραφο τρέχουσας διεργασίας
    - Επιστρέφει αριθμό παιδιού στον πατέρα και 0 στο παιδί
  - waitpid(): αναμονή μέχρι να τερματίσει ένα παιδί
    - Επιστρέφει αριθμό διεργασίας και κατάσταση εξόδου
  - execve(): αντικαθιστά τον κώδικα μιας διεργασίας
  - exit(): τερματισμός τρέχουσας διεργασίας

Διαχείριση διεργασιών (2 από 3)

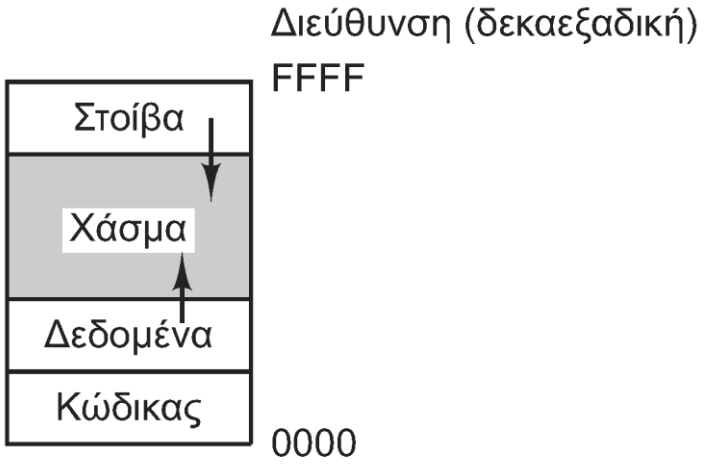
```
#define TRUE 1

while (TRUE) {                                  /* επανάληψη συνεχώς */
    type_prompt();                              /* εκτύπωση του προτρεπτικού μηνύματος */
                                                    /* στην οθόνη */
    read_command(command, parameters);          /* ανάγνωση της εισόδου από το τερματικό */
    if (fork() != 0) {                          /* δημιουργία θυγατρικής διεργασίας */

        /* κώδικας γονικής διεργασίας */
        waitpid(–1, &status, 0);                /* αναμονή ολοκλήρωσης θυγατρικής */
    } else {
        /* κώδικας θυγατρικής διεργασίας */
        execve(command, parameters, 0);         /* εκτέλεση της εντολής */
    }
}
```

- Παράδειγμα: ένας απλός φλοιός (shell)
  - Ανάγνωση εντολής, δημιουργία παιδιού για εκτέλεση
  - Ο πατέρας περιμένει το παιδί και επαναλαμβάνει

Διαχείριση διεργασιών (3 από 3)



- Χάρτης μνήμης διεργασίας στο UNIX
  - Κώδικας, δεδομένα και στοίβα
- Παράμετροι εκτέλεσης διεργασιών (argc, argv, envp)
  - argc, argv: πλήθος παραμέτρων και δείκτες σε αυτές
  - envp: δείκτες σε μεταβλητές περιβάλλοντος

Διαχείριση αρχείων (1 από 4)

Κλήση	Περιγραφή δράσης
fd = open(αρχείο, τρόπος, ...)	Ανοίγει ένα αρχείο για ανάγνωση, εγγραφή, ή και τα δύο
s = close(fd)	Κλείνει ένα ανοιχτό αρχείο
n = read(fd, buffer, nbytes)	Διαβάζει δεδομένα από ένα αρχείο και τα τοποθετεί σε προσωρινή μνήμη
n = write(fd, buffer, nbytes)	Διαβάζει δεδομένα από μια προσωρινή μνήμη και τα αποθηκεύει σε ένα αρχείο
position = lseek(fd, απόσταση, whence)	Μετακινεί το δείκτη αρχείου
s = stat(όνομα, &buf)	Διαβάζει τις πληροφορίες κατάστασης ενός αρχείου

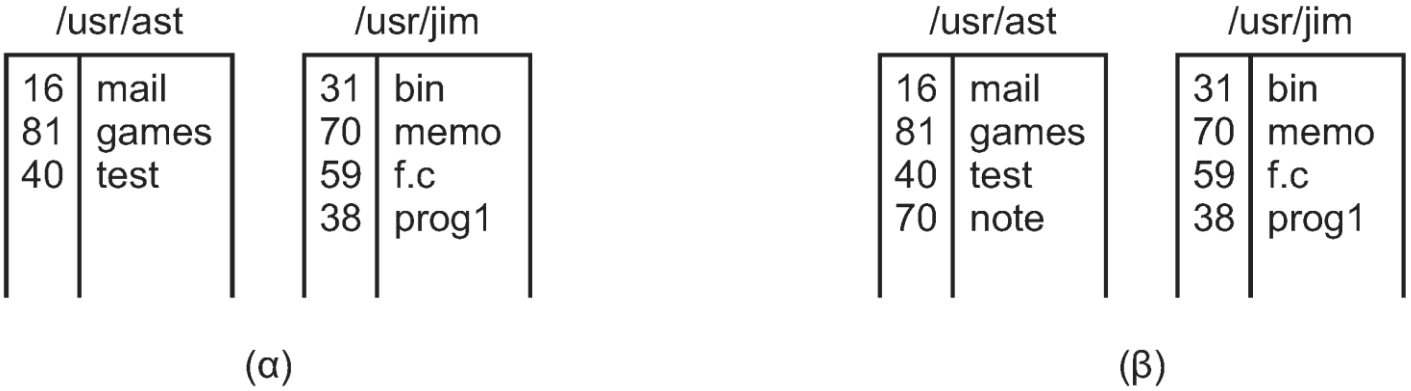
- Κλήσεις διαχείρισης αρχείων
  - open(): άνοιγμα ή δημιουργία αρχείου
  - close(): κλείσιμο αρχείου
  - read()/write(): ανάγνωση/εγγραφή αρχείου
  - lseek(): μετακίνηση σημείου ανάγνωσης/εγγραφής
  - stat(): ανάγνωση μεταδεδομένων αρχείου

Διαχείριση αρχείων (2 από 4)

Κλήση	Περιγραφή δράσης
s = mkdir(όνομα, κατάσταση)	Δημιουργεί ένα νέο κατάλογο
s = rmdir(όνομα)	Διαγράφει έναν κενό κατάλογο
s = link(όνομα1, όνομα2)	Δημιουργεί μια νέα καταχώριση όνομα2 η οποία δείχνει στο όνομα1
s = unlink(όνομα)	Διαγράφει μια καταχώριση καταλόγου
s = mount(ειδική, όνομα, σημαία)	Αναρτά ένα σύστημα αρχείων
s = umount(ειδική)	Αποαναρτά ένα σύστημα αρχείων

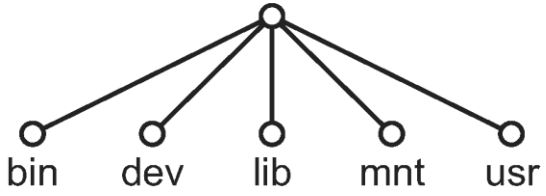
- Διαχείριση καταλόγων
  - mkdir(): δημιουργία νέου καταλόγου
  - rmdir(): διαγραφή καταλόγου (πρέπει να είναι κενός)
  - link(): δημιουργία συνδέσμου προς αρχείο/κατάλογο
  - unlink(): διαγραφή συνδέσμου ή αρχείου
  - (u)mount(): ανάρτηση/απομάκρυνση συστήματος αρχείων

Διαχείριση αρχείων (3 από 4)

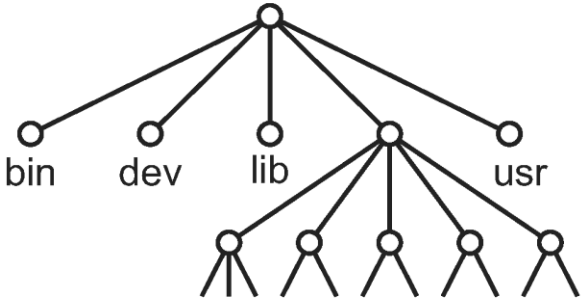


- Λειτουργία των συνδέσμων (ή συνδέσεων)
  - Παράδειγμα: link(“/usr/jim/memo”,“/usr/ast/note”);
  - Στον /usr/ast εμφανίζεται το αρχείο /usr/jim/memo
  - Στο UNIX κάθε αρχείο αντιπροσωπεύεται από έναν κόμβο i
  - Η link() εισάγει δείκτη προς υπάρχοντα κόμβο i
  - Το αρχείο διαγράφεται όταν γίνει unlink() από παντού

Διαχείριση αρχείων (4 από 4)



(α)



(β)

- Λειτουργία της ανάρτησης
  - mount(“/dev/hda”,“/mnt”,0);
  - Ανάρτηση του σκληρού δίσκου /dev/hda κάτω από το /mnt
  - Το σύστημα αρχείων του δίσκου είναι ορατό
  - Το σύστημα ξεκινάει με ένα ριζικό σύστημα αρχείων
  - Στη συνέχεια αναρτώνται πρόσθετα συστήματα αρχείων

Διάφορες κλήσεις

Κλήση	Περιγραφή δράσης
s = chdir(όνομακαταλόγου)	Αλλάζει τον κατάλογο εργασίας
s = chmod(όνομα, κατάσταση)	Αλλάζει τα bit προστασίας ενός αρχείου
s = kill(pid, σήμα)	Στέλνει σήμα σε μια διεργασία
seconds = time(&seconds)	Υπολογίζει το χρόνο που έχει περάσει από την 1/1/1970

- chdir(): αλλαγή καταλόγου εργασίας
- chmod(): αλλαγή προνομίων πρόσβασης
  - Read/write/execute για user/group/others
- kill(): αποστολή σήματος σε διεργασία
  - Όσα δεν συλλαμβάνονται σκοτώνουν τον παραλήπτη
- time(): επιστρέφει την τρέχουσα ώρα
  - Χρονικό διάστημα από μια σταθερή χρονική στιγμή

Windows και UNIX

- Το μοντέλο προγραμματισμού Windows διαφέρει
  - Επικεντρώνεται στα γεγονότα από το περιβάλλον
- Η βασική διεπαφή είναι το Win32 API
  - Χρησιμοποιείται σε πολλές εκδόσεις Windows
  - Η ανάγκη συμβατότητας κάνει τη διεπαφή ασυνεπή
  - Τα Windows έχουν πολλά API και χιλιάδες κλήσεις
- Ορισμένες διαφορές των Windows από το UNIX
  - Η δημιουργία νέας διεργασίας θέλει ένα μόνο βήμα
  - Παρέχεται μία κλήση για αναμονή πολλών συμβάντων

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Δομή ΛΣ

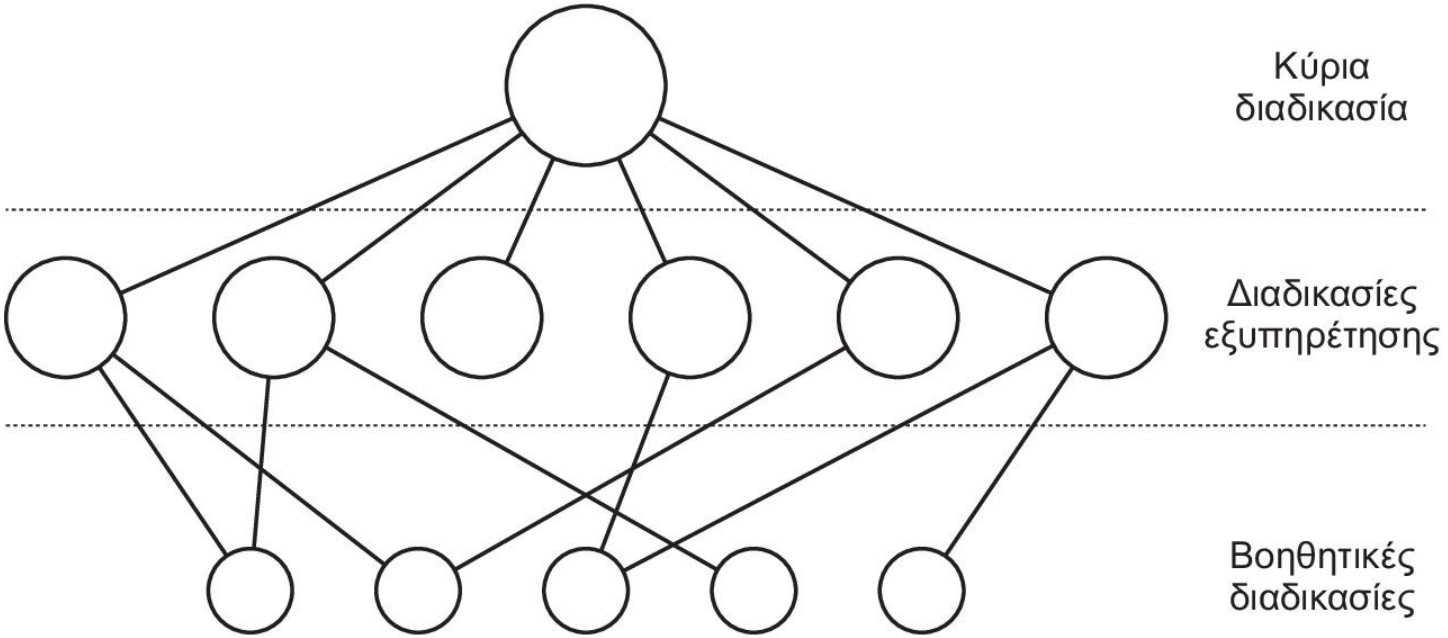
Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Δομή ΛΣ (1 από 10)

- Μονολιθικά συστήματα
  - Το ΛΣ είναι ένα τεράστιο πρόγραμμα
    - Εκτελείται όλο σε κατάσταση πυρήνα
  - Κάθε διαδικασία μπορεί να καλέσει κάθε άλλη
    - Τα πάντα είναι ορατά σε όλους (δομές, διαδικασίες)
  - Στοιχειώδης οργάνωση κώδικα
    - Η είσοδος στο ΛΣ γίνεται με μια παγίδα
    - Κύριο πρόγραμμα που καλεί αντίστοιχη διαδικασία
    - Διαδικασίες που εξυπηρετούν τις κλήσεις συστήματος

Δομή ΛΣ (2 από 10)



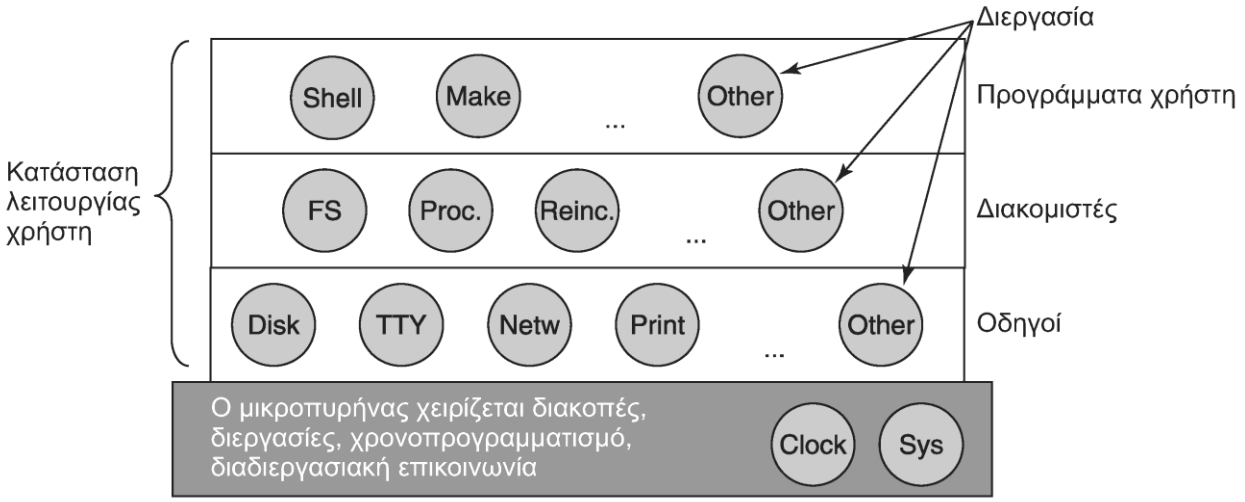
- Προσθήκη κώδικα κατά τη λειτουργία
  - Δυναμικά φορτώσιμες βιβλιοθήκες
  - Δυναμικά φορτώσιμοι οδηγοί συσκευών

Δομή ΛΣ (3 από 10)

Επίπεδο	Λειτουργία
5	Χειριστής
4	Προγράμματα χρηστών
3	Διαχείριση εισόδου/εξόδου
2	Επικοινωνία χειριστή-διεργασίας
1	Διαχείριση μνήμης και τυμπάνου
0	Εκχώρηση επεξεργαστή και πολυπρογραμματισμός

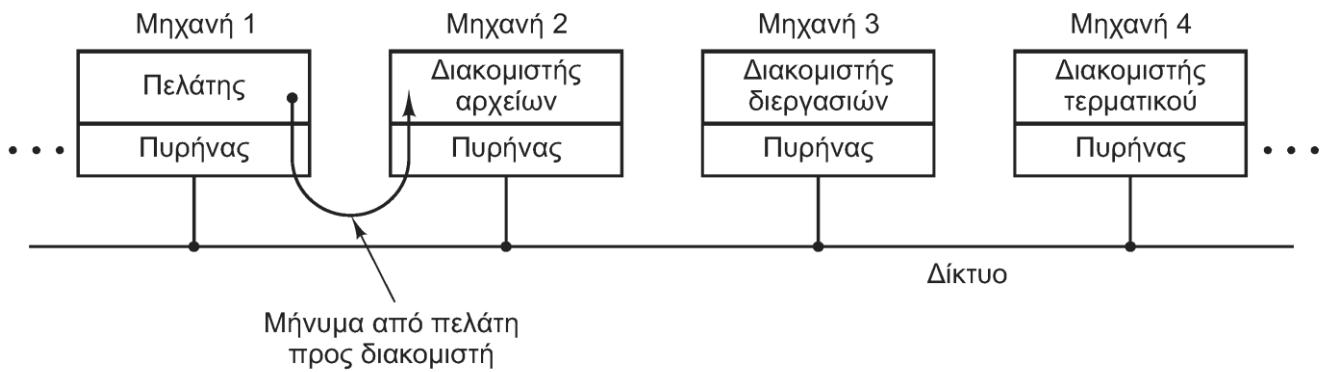
- Πολυεπίπεδα συστήματα
  - Ιδέα του E.W. Dijkstra που εφαρμόστηκε στο THE
  - Κάθε επίπεδο παρέχει υπηρεσίες στα παραπάνω
  - Γενικεύθηκε στο σύστημα MULTICS
    - Οργάνωση σε ομόκεντρους δακτυλίους
    - Συνδυασμός δόμησης σε επίπεδα και προστασίας

Δομή ΛΣ (4 από 10)



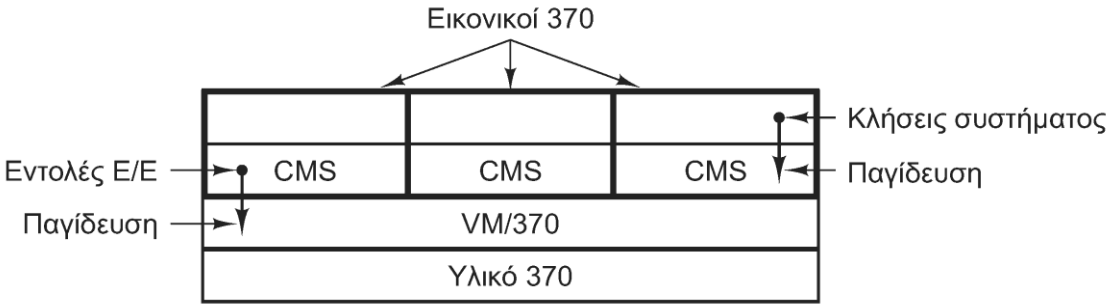
- Μικροπυρήνες
  - Ελάχιστο σύνολο λειτουργιών σε κατάσταση πυρήνα
    - Διακοπές, νέες διεργασίες, χρονοπρογραμματισμός, IPC
  - Οι υπόλοιπες εκτελούνται σε κατάσταση χρήστη
  - Επικοινωνία μέσω μηνυμάτων ή IPC

Δομή ΛΣ (5 από 10)



- Μοντέλο πελάτη-εξυπηρετητή
  - Εξυπηρετητές: παρέχουν υπηρεσίες
  - Πελάτες: ζητούν υπηρεσίες από τους εξυπηρετητές
  - Επικοινωνία μέσω μεταβίβασης μηνυμάτων
    - Απλουστεύει την κατανομή υπηρεσιών σε δίκτυο

Δομή ΛΣ (6 από 10)

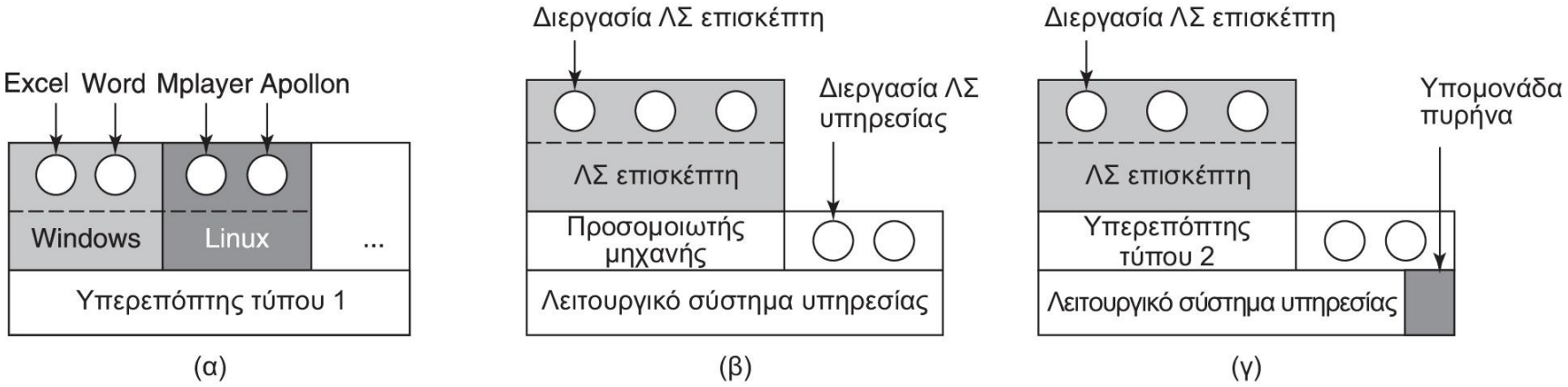


- Εικονικές μηχανές (VM)
  - Ξεκίνησε με το VM/370 και επιβιώνει στο z/VM
  - Απλός πυρήνας που εκτελείται πάνω στο υλικό
  - Παρέχει την εικόνα πολλών πανομοιότυπων μηχανών
    - Παγίδευση και εκτέλεση των προνομιούχων εντολών
    - Η παγίδευση υλοποιείται με ειδικό υλικό
    - Εκτέλεση ΛΣ δέσμης ή διαλογικών ΛΣ ενός χρήστη

Δομή ΛΣ (7 από 10)

- Η επιστροφή: εικονικοποίηση χωρίς υλικό
  - Τροποποίηση κώδικα ώστε να καλεί υπερεπόπτη
    - VMware: δυαδική μετάφραση κώδικα
    - XEN: τροποποίηση κώδικα λειτουργικού
  - Εκτέλεση κανονικών ΛΣ όπως Windows και Linux
    - Τα κλασικά VMM εκτελούσαν απλούστερα ΛΣ
- Η ιστορία επαναλαμβάνεται: ειδικό υλικό
  - Υποστήριξη VM από επεξεργαστή, κάρτες δικτύου

Δομή ΛΣ (8 από 10)



- Δύο τύποι υπερεπόπτη
  - Τύπου 1: εκτελείται απευθείας πάνω στο υλικό
    - Όλα τα εικονικά ΛΣ είναι ισότιμα
  - Τύπου 2: εκτελείται μέσα σε ένα ΛΣ
    - Επισκέπτης (guest) και υπερεπόπτης (host)
    - Πιθανόν με πρόσθετη υποστήριξη από πυρήνα

Δομή ΛΣ (9 από 10)

- Εξωπυρήνες
  - Διαμέριση πόρων ανάμεσα σε εικονικές μηχανές
    - Κάθε χρήστης παίρνει μέρος του δίσκου και της μνήμης
  - Κάθε μηχανή γνωρίζει ότι έχει μέρος των πόρων
    - Στην εικονικοποίηση νομίζει ότι έχει όλη τη μηχανή
  - Πολύ απλούστερη υλοποίηση του εξωπυρήνα
    - Όχι μετάφραση εικονικών σε φυσικούς πόρους
    - Αλλά απαιτεί ΛΣ που να συμπεριφέρεται ανάλογα

Δομή ΛΣ (10 από 10)

- Unikernels (μονοπυρήνες;)
  - Ιδέα που ξεκινά από την εικονικοποίηση
  - Συνήθως ένα VM τρέχει λίγα προγράμματα
  - Γιατί η εικονική μηχανή να τρέχει ολόκληρο ΛΣ;
  - Το unikernel περιέχει μόνο τα απαραίτητα
    - Τμήματα του ΛΣ (π.χ., όχι GUI)
    - Ελάχιστο σύνολο βιβλιοθηκών
  - Χρειάζεται καλά δομημένο λειτουργικό

Η γλώσσα C

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής

Η γλώσσα C (1 από 4)

- Από την Java στην C σε λίγα λεπτά
  - Η Java μοιάζει αρκετά στη σύνταξη με τη C
  - Απλοί τύποι δεδομένων (int, float, double, …)
  - Δομές (structs) από απλούστερα στοιχεία
  - Ενώσεις (unions) - δεν υπάρχουν αλλού!
  - Πίνακες απλούστερων στοιχείων
  - Κοινές εντολές ελέγχου (if, switch, for, while)
  - Απλή στη μεταγλώττιση και κοντά στη μηχανή

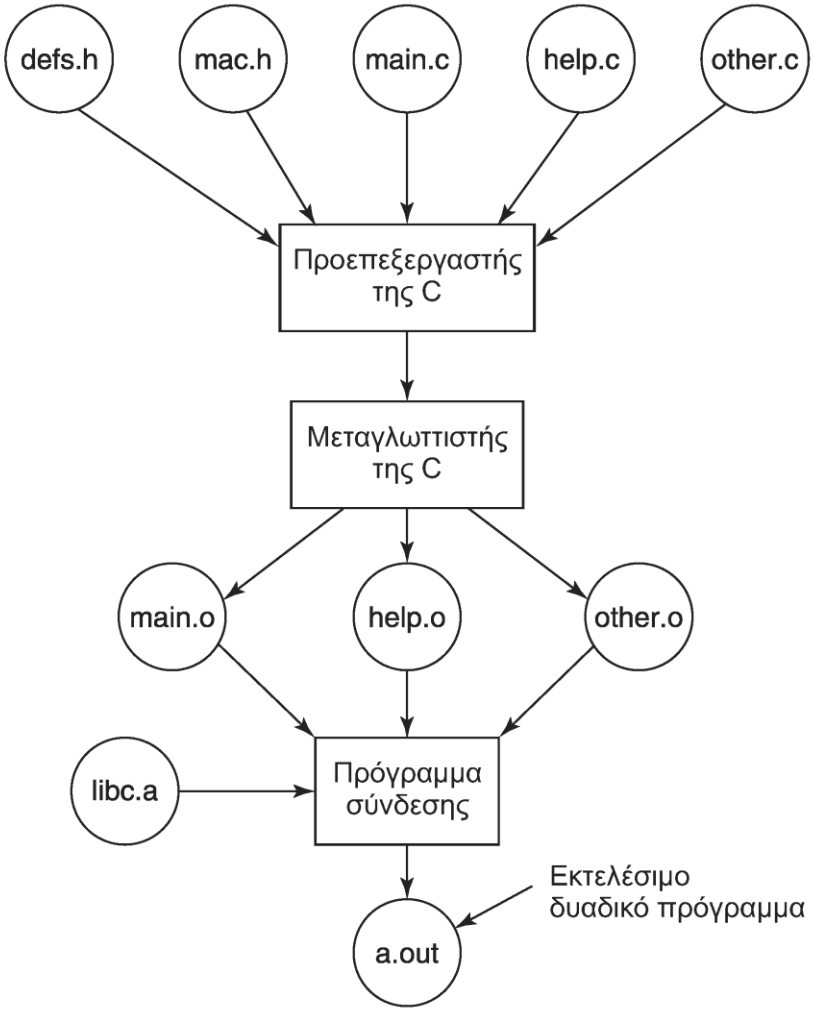
Η γλώσσα C (2 από 4)

- Οι μεγάλες διαφορές
  - Δεν υπάρχουν αντικείμενα
  - Δυνατότητα χρήσης δεικτών στη μνήμη
    - Θεωρητικά έχουν τύπους (π.χ. σε ακέραιο)
    - Μπορούμε να κάνουμε και πράξεις!
  - Η C δεν κάνει αυτόματη διαχείριση μνήμης
    - Απαιτείται ρητή δέσμευση και αποδέσμευση μνήμης
  - Η C είναι πολύ ισχυρή, αλλά και πολύ επικίνδυνη

Η γλώσσα C (3 από 4)

- Αρχεία κεφαλίδες
  - Δηλώσεις και ορισμοί τύπων / μακροεντολών
  - Αντικαταστάσεις από τον προεπεξεργαστή C
    - Αντικατάσταση κειμένου, μεταγλώττιση υπό συνθήκη
- Μεταγλώττιση και σύνδεση
  - Ο μεταγλωττιστής καλεί τον προεπεξεργαστή
  - Η έξοδος του μεταγλωττιστή είναι αντικειμενικό αρχείο
  - Τα αρχεία συνδέονται (στατικά/δυναμικά) σε εκτελέσιμα
  - Χρήση make για μερική μεταγλώττιση

Η γλώσσα C (4 από 4)



ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Τέλος Ενότητας #1

Μάθημα: Λειτουργικά Συστήματα, **Ενότητα # 1:** Εισαγωγή

Διδάσκων: Γιώργος Ξυλωμένος, **Τμήμα:** Πληροφορικής