Προγραμματισμός

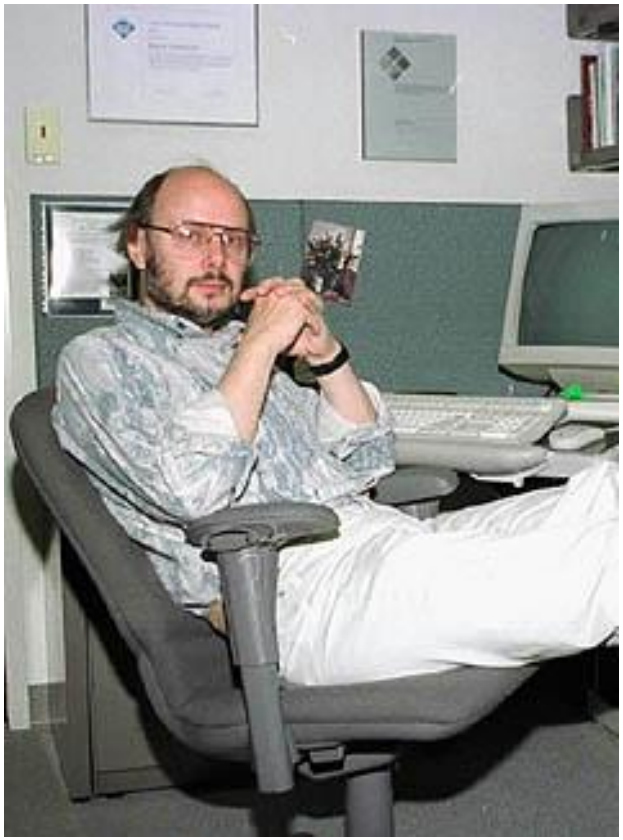
Αλγόριθμοι και Προγράμματα

Άνθρωπος - Μηχανή

- Χρήση υπολογιστή αν:
  - Έχουμε πολλά δεδομένα/αποτελέσματα
  - Μέθοδος επίλυσης πολύπλοκη ή βαρετή για τον άνθρωπο.
  - Χρησιμοποιούμε την ίδια μέθοδο πολλές φορές.
- Ο Η/Υ εκτελεί πράξεις γρηγορότερα από τον άνθρωπο.
- Ο Η/Υ μπορεί να εκτελέσει με συνέπεια και πειθαρχεία μια λογική σειρά εντολών.
- Ο Η/Υ δεν κουράζεται ούτε βαριέται

Η γλώσσα C++

- Η γλώσσα C αναπτύχθηκε το 1983 από τον Bjarne Stroustrup στα Bell Labs, ως βελτίωση της ήδη υπάρχουσας γλώσσας προγραμματισμού C, και αρχικά ονομάστηκε "C with Classes", δηλαδή C με Κλάσεις.
- Γλώσσα γενικού σκοπού και μια από τις πιο διαδεδομένες γλώσσες υψηλού επιπέδου, αλλά για «μεγαλύτερα» προγράμματα
- Η γλώσσα ορίστηκε, το 1998, με το πρότυπο ISO/IEC 14882:1998.



Πρόδρομος: Η γλώσσα C

- Η γλώσσα C αναπτύχθηκε το 1972 από τον Dennis Ritchie στα AT&T (Bell) Laboratories.
  - Ήταν η τρίτη προσπάθεια (οι άλλες δύο ήταν οι A και οι B)
- Γλώσσα γενικού σκοπού και μια από τις πιο διαδεδομένες γλώσσες υψηλού επιπέδου
- Σχεδιάστικε και χρησιμοποιήθηκε για την κατασκευή του λειτουργικό σύστημα UNIX

Γιατί;

- Οι περισσότερες τεχνικές (ακόμη και πολλές μηχανικές) διεργασίες πλέον απαιτούν συγγραφή λογισμικού
- Λογισμικό υπάρχει και πέραν του προσωπικού μας ΗΥ

Ships

- Design
- Construction
- Management

- Monitoring
- Engine
- Hull design
- Pumps

Aircraft

- Communication
- Control
- Display

- Signal processing
- “Gadget” control
- Monitoring

Phones

- Voice quality
- User interfaces
- Billing
- Mobility

- Switching
- Reliability
- Provisioning
- Images

Phone Apps

- Android & iPhone use C/C++

Energy

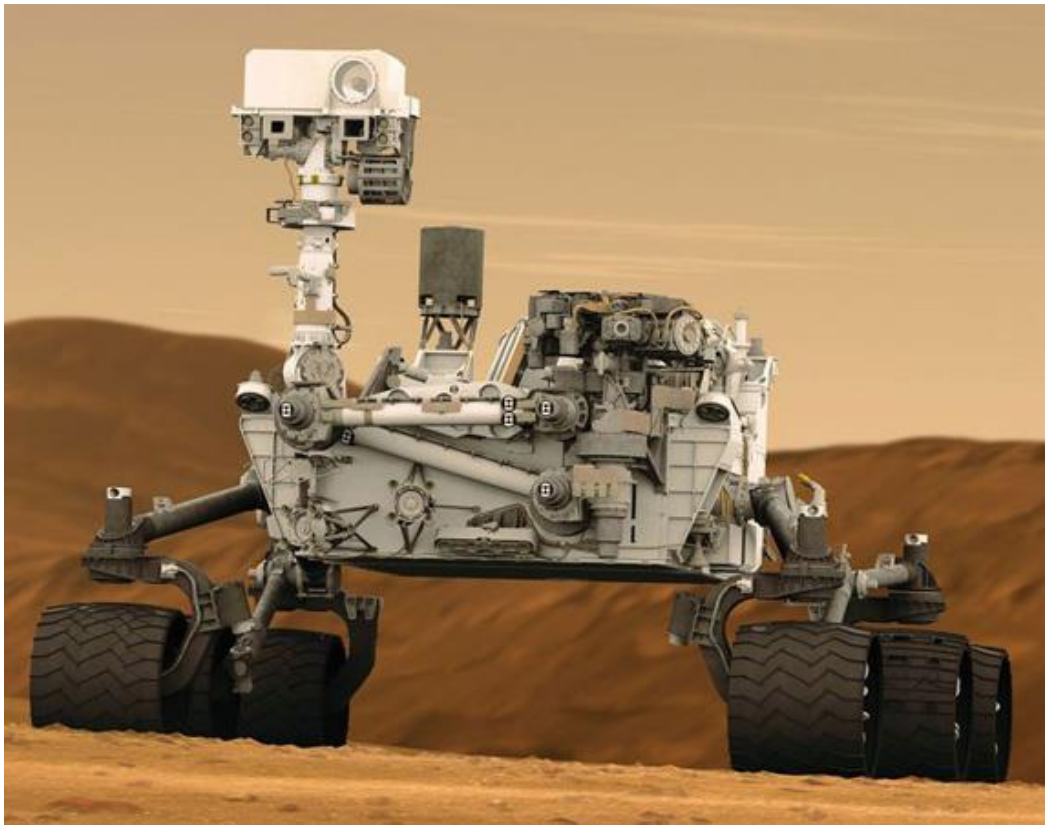
- Control
- Monitoring
- Analysis
- Design

- Communications
- Visualization
- Manufacturing

PC/workstation

- There’s a lot more to computing than games, word processing, browsing, and spreadsheets!

Where is C++ Used?



Mars rovers, animation, graphics, Photoshop, GUI, OS, compilers, slides, chip design, chip manufacturing, semiconductor tools, etc.

See www.research.att/~bs/applications.html

Α Μέρος

Εισαγωγή στον προγραμματισμό

Υλικό: Hardware

Δευτερεύουσα μνήμη


Συσκευές εισόδου



RAM
(κύρια μνήμη)

CPU

HY
(κεντρική μονάδα επεξεργασίας)

Συσκευές εξόδου

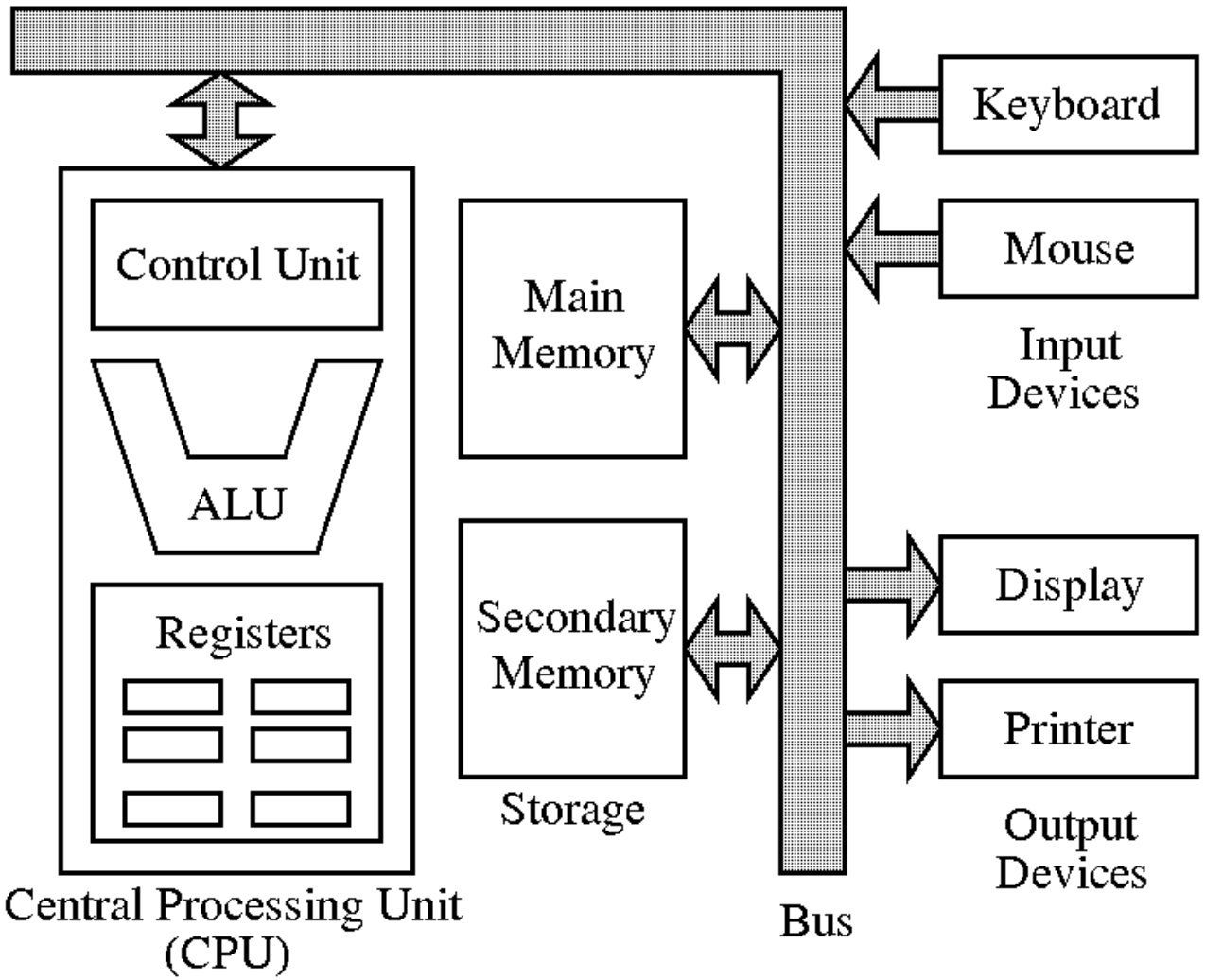


Οθόνες αφής

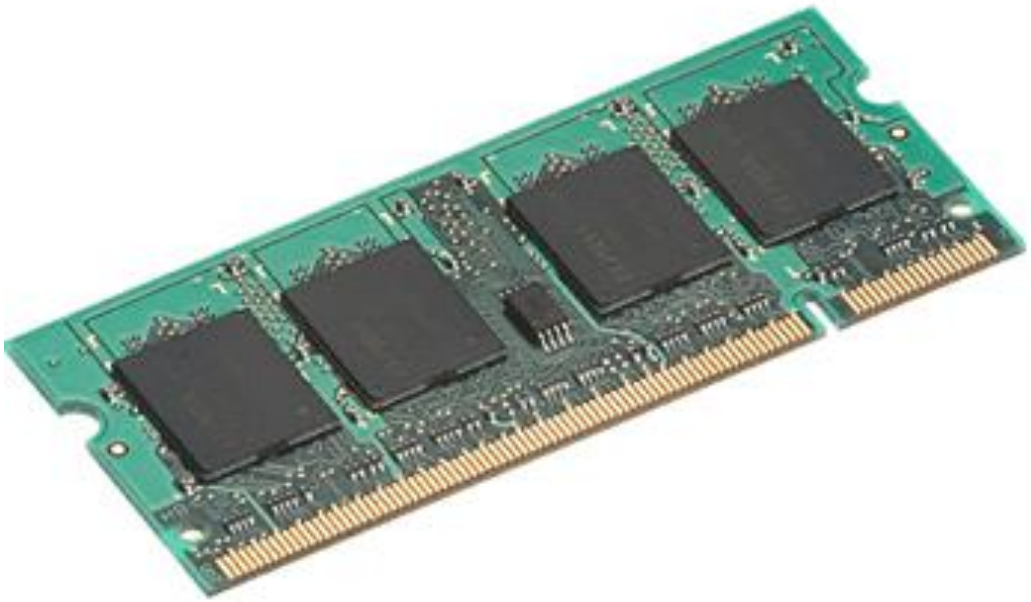
Ανατομία ενός ΗΥ

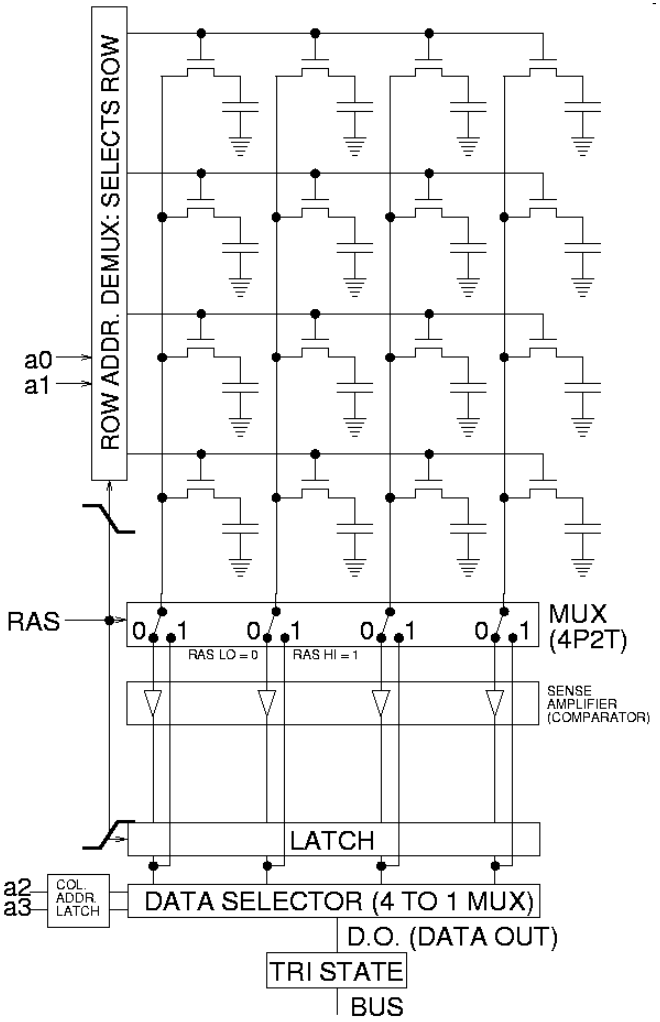
- Μέσω των **συσκευών εισόδου – input** δεδομένα και προγράμματα εισάγονται από τον έξω κόσμο στον Η/Υ.
- Μέσω των **συσκευών εξόδου – output** δεδομένα και ενέργειες του Η/Υ διατίθενται στον εξωτερικό κόσμο.
- Η **μνήμη** είναι μια συλλογή από κελιά (cells) το καθένα από τα οποία έχει μια μοναδική φυσική διεύθυνση. Υπάρχει η κύρια και η δευτερεύουσα μνήμη.
- Η **κεντρική μονάδα επεξεργασίας** δέχεται εντολές και δεδομένα. Τα δεδομένα επεξεργάζονται σύμφωνα με τις εντολές που έχουν δοθεί και τα αποτελέσματα διαβιβάζονται στις κατάλληλες μονάδες ή στη μνήμη.

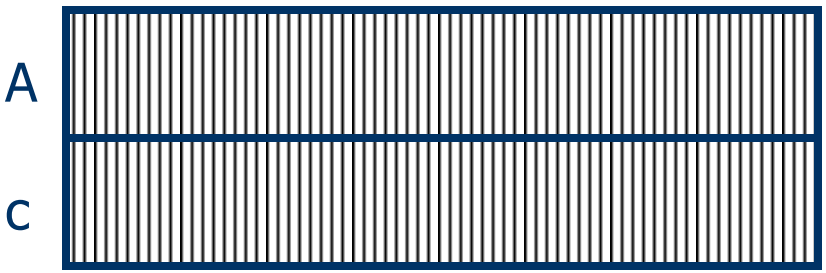
Οργάνωση ενός ΗΥ



Μνήμη





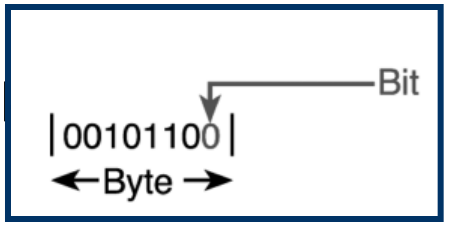


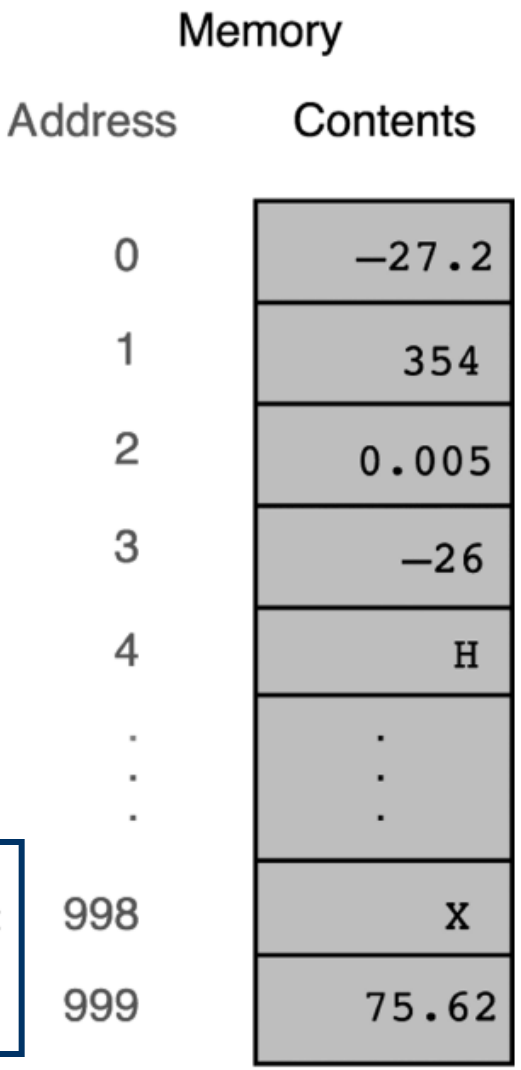
Addr #42

Addr #43

Μνήμη

- Είναι ο χώρος όπου αποθηκεύει τα δεδομένα και τις εντολές που επεξεργάζεται.
- Ελάχιστη θέση μνήμης που μας δίνει η γλώσσα C = 1 byte (λόγω memory alignment στα αλήθεια είναι 1 word, 4/8 bytes, βλ. «Οργάνωση Υπολογιστών».
- Δυαδική αναπαράσταση, δυαδικά ψηφία {0, 1}, ή 8 bits (**bi**nary digi**t**).
  - Προσωρινή (πρωτεύουσα, γρήγορη, σε ηλεκτρονικό κύκλωμα)
  - Μόνιμη (δευτερεύουσα, αργή, μέλη)





Λογισμικό / Software

- Λογισμικό Συστήματος (System Software):
  - Περιλαμβάνει το σύνολο των προγραμμάτων που ελέγχουν διαχειρίζονται και συντονίζουν τους πόρους του Η/Υ
  - Λειτουργεί σε καθεστός ανεξαρτησίας από συγκεκριμένες εφαρμογές.
  - Η λειτουργία του δεν είναι άμεσα αντιληπτή από τον απλό χρήστη.
- Λογισμικό Εφαρμογών (Application Software):
  - Περιλαμβάνει προγράμματα που επιτρέπουν την εκτέλεση συγκεκριμένων εργασιών των χρηστών π.χ την δημιουργία κειμένων, την σχεδίαση γραφικών, την οργάνωση δεδομένων κλπ.

Προγραμματισμός και Αλγόριθμοι

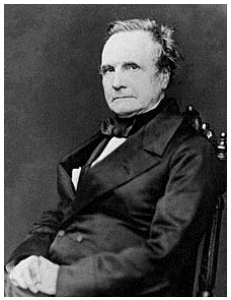
- Γλώσσα Μηχανής: η μόνη γλώσσα που πραγματικά καταλαβαίνει ο υπολογιστής
- Γλώσσες Προγραμματισμού
  - Χαμηλού επιπέδου, πιο κοντά στη γλώσσα μηχανής
  - Μεσαίου
  - Υψηλού, πιο κοντά στην φυσική (μας) γλώσσα
- Ο προγραμματισμός είναι πιο αυτονόητος στις υψηλού επιπέδου γλώσσες, τα προγραμμάτα όμως εκτελούνται βραδύτερα σε σύγκριση με τις χαμηλού επιπέδου γλώσσες.
  - Πολλά όμως εξαρτώνται και από τον **προγραμματιστή**.
  - Οι μεταφραστές έχουν εξελιχθεί
  - Η γλώσσα C++ μπορεί να παράξει ταχύτατα προγράμματα

Οικογένειες Γλωσσών Προγραμματισμού

- Μερικές σημαντικές κλάσεις:
- Διαδικαστικές (Procedural) : *C, Pascal, Fortran*
  - Παραδοσιακός τρόπος προγραμματισμού
  - Σειρά εντολών - ρουτίνες
- Συναρτησιακές (Functional) : *Lisp, Mathematica*
  - Σειρά εντολών
  - Οι συναρτήσεις – συμβολισμοί είναι αντίστοιχοι με τα μαθηματικά
- Δηλωτικές (Declarative) : *Prolog, XML*
  - Αποτελούνται από σειρά ορισμούς και δηλώσεις
  - Λογικός Προγραμματισμός (Logic Programming)
  - Προγραμματισμός με βάση Περιορισμούς (Constraint Programming)

- Αντικειμενοστραφής Προγραμματισμός
- Παράλληλος Προγραμματισμός
- Διαδικτυακός Προγραμματισμός

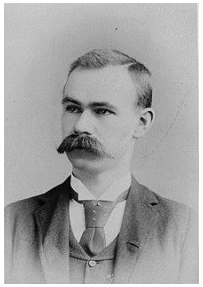
Πρώτες γλώσσες προγραμματισμού



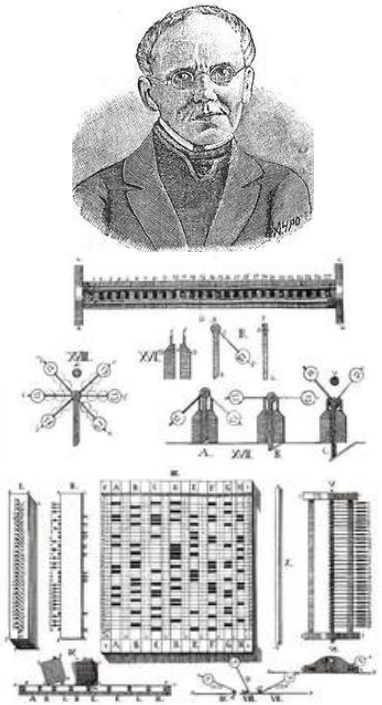
https://en.wikipedia.org/wiki/Charles_Babbage

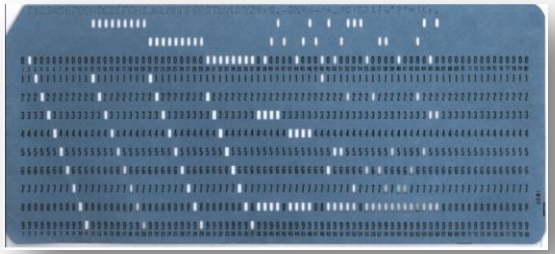
https://en.wikipedia.org/wiki/Ada_Lovelace

https://en.wikipedia.org/wiki/Herman_Hollerith

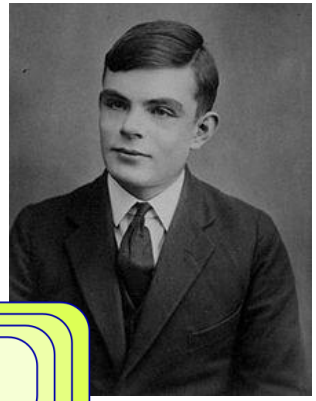


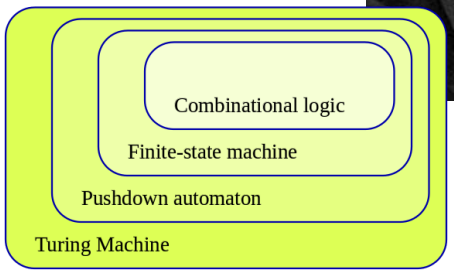
https://en.wikipedia.org/wiki/Semen_Korsakov

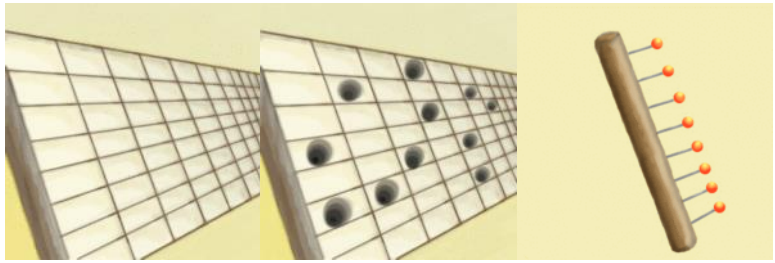




https://en.wikipedia.org/wiki/Alan_Turing







Γλώσσες προγραμματισμού

- Γλώσσα Μηχανής- Machine Language:
  - Κωδικοποίηση εντολών σε δυαδική μορφή (0,1).
  - Έχει 1-1 αντιστοιχία με την assembly (κατανοητή από ανθρώπους)
- Διακρίνεται (διακρινόταν) για την ταχύτητα εκτέλεσης των εντολών της και την βέλτιστη χρήση της κύριας μνήμης.
  - Είναι δύσκολη στην χρήση της, στον εντοπισμό και διόρθωση λαθών

Γλώσσες προγραμματισμού

- Γλώσσες υψηλού επιπέδου
  - Βρίσκονται πιο κοντά στις ανθρώπινες γλώσσες.
  - Χρησιμοποιούν ένα μικρό σύνολο από αγγλικές λέξεις.
- Για να εκτελεστεί ένα τέτοιο πρόγραμμα γίνεται χρήση μεταγλωττιστών (Compilers)
  - Παραδείγματα: Fortran, Basic, Pascal, C/C++.

Διερμηνευμένες Γλώσσες (Interpreted)

Αλγόριθμος

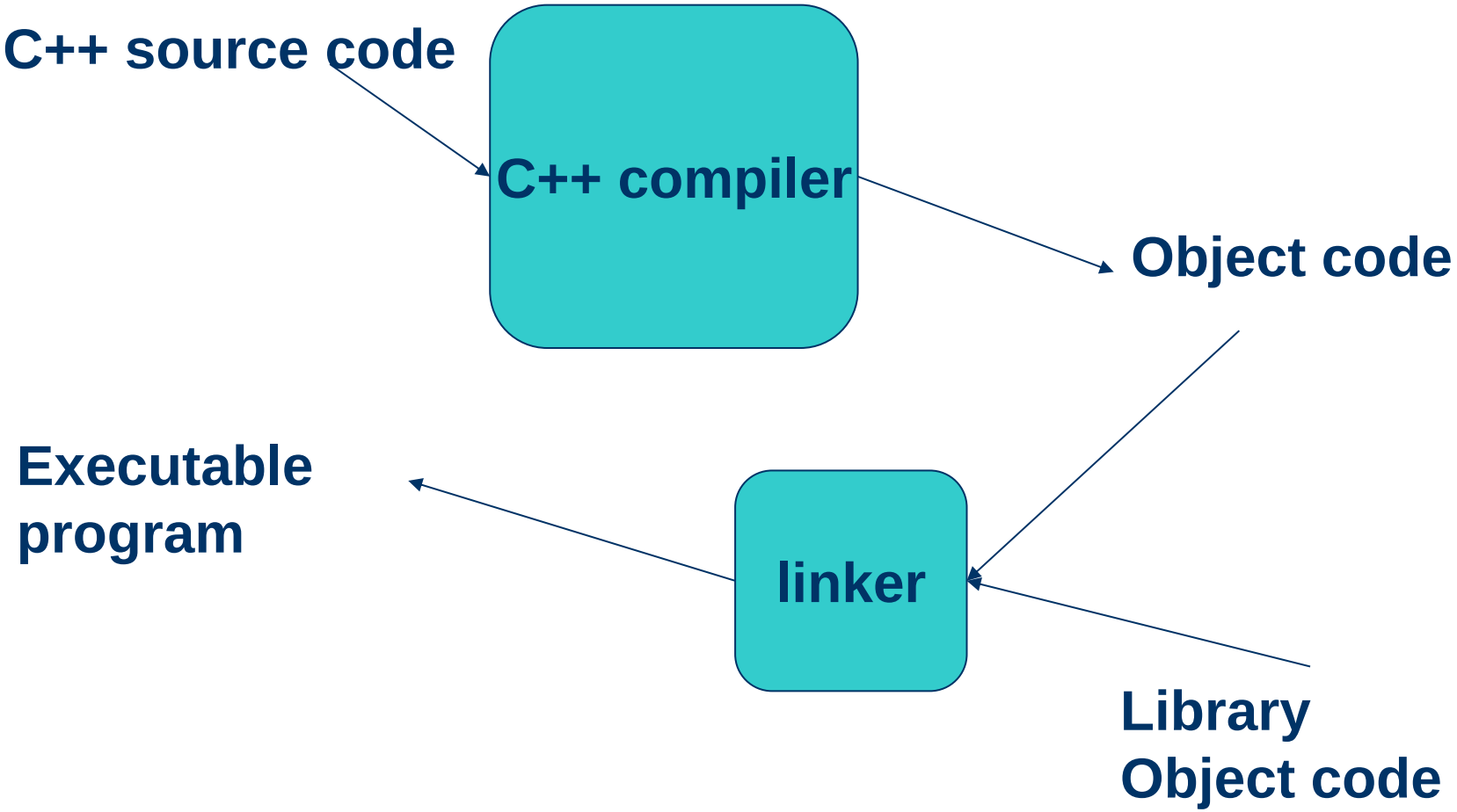
|

Γραμμή εντολών (κώδικας) (π.χ. “move arm”, “show image”, “run script A1”, “move west”)

|

Διερμηνέας (Interpreter), ανάλυση και εκτέλεση εντολής

Συγγραφή, μεταγλώττιση και εκτέλεση ενός προγράμματος σε γλώσσα υψηλού επιπέδου



Δημιουργία Προγράμματος

Αλγόριθμος (π.χ. Bubble Sort)

|

Γλώσσα Προγραμματισμού (BubbleSort.c)

|----------compiler

Ενδιάμεσος Κώδικας (object code) (BubbleSort.o)

| ----------linker

Εκτελέσιμο (Executable) (BubbleSort.exe)

Δημιουργία Προγράμματος

Αλγόριθμος

|

Γλώσσα Προγραμματισμού

|----------Συντακτικά Λάθη

Ενδιάμεσος Κώδικας (object code)

| ---------- Λάθη Συμβόλων

Εκτελέσιμο (Executable) ---------- Λογικά Λάθη

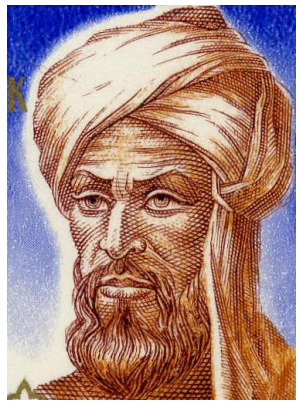
Run Time Errors

Προγραμματιστική επίλυση προβλήματος

- Περιγραφή του προβλήματος. Προσδιορισμός απαιτήσεων.
- Ανάλυση του προβλήματος.
  - Προσδιορισμός εισόδων, εξόδων.
  - Προσδιορισμός της λύσης.
- Σχεδίαση της λύσης του προβλήματος.
  - Ανάλυση **αλγορίθμου**.
  - Σχεδιασμός διαγράμματος ροής.
  - Δημιουργία ψευδοκώδικα.
- Κωδικοποίηση του αλγορίθμου σε γλώσσα προγραμματισμού.
- Δοκιμή, έλεγχος και διόρθωση λαθών.
- Συντήρηση και τεκμηρίωση του προγράμματος

Αλγόριθμος

- Αλγόριθμος:
  - ένα σύνολο εκτελέσιμων και σαφών εντολών που κατευθύνει μία διαδικασία που τερματίζει
  - Ένα διατεταγμένος και περασμένο σύνολο εντολών για τον υπολογισμό μιας συνάρτησης
  - Η λέξη αλγόριθμος (ή αλγορισμός) προέρχεται από το όνομα του Πέρση μαθηματικού, γεωγράφου και αστρονόμου, Muḥammad ibn Musa al-Khwarizmi, από τον οποίο προέρχεται και η λέξη Άλγεβρα επίσης.



Αλγόριθμοι

- Μια ακολουθία από βήματα/ενέργειες που είναι:
  - Καλώς (σαφώς) ορισμένα
  - Αποτελεσματικά (μπορούν να εκτελεστούν)
  - Πεπερασμένα (τερματισμός)
  - Συνήθως δέχονται δεδομένα
- Παραδείγματα αλγορίθμων
  - Υπολογισμού ημερομηνίας Πάσχα
  - Υπολογισμός ριζών
  - Πολλ/σμός – Αντιστροφής πινάκων
  - Αναζήτησης και Ταξινόμησης
  - Εύρεσης συντομότερου μονοπατιού
  - Πρόβλεψης
  - Ανάλυσης ιατρικών εικόνων
  - Δρομολόγησης πακέτων στο Internet

Αλγόριθμοι

- Απαιτήσεις - Πρόβλημα – Προδιαγραφές
- **Σχεδίαση - Ορθότητα**
- Πολυπλοκότητα (με βάση το μέγεθος των δεδομένων N) (προαιρετική για το μάθημα μας)
  - Υπολογιστικό Κόστος
  - Κόστος σε Μνήμη
- Βελτιστοποίηση (προαιρετική για το μάθημα μας)

- **Υλοποίηση**

Παράδειγμα (1/3)

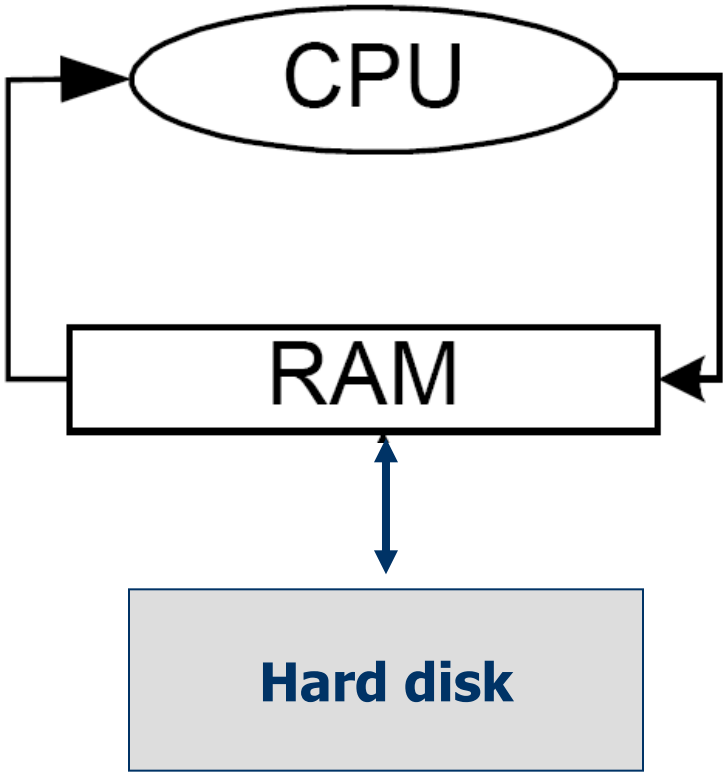
- Πρόβλημα: με είσοδο τη λίστα φοιτητών όλου του τμήματος, τύπωσε τα ονομάτα των φοιτητών του πρώτου έτους

STUDENTS.DAT

Επίθετο	Όνομα	Ηλεκτρονικό Ταχυδρομείο	Έτος	Τηλέφωνο
Davis	Albert	none	1	1668-78-9226
Crane	Amory	none	2	2689-48-8430
Schakowsky	Anibal	none	3	3652-58-7355
Kirk	Anne	rep.kirk@volcano.com	1	4623-87-0203
Weller	Anthony	none	1	1780-52-0498
Costello	Barbara	none	3	1599-98-7962
Biggert	Barney	none	2	1058-86-1065
Hastert	Baron	speaker@volcano.com	1	1075-45-4923

Παράδειγμα (2/3)

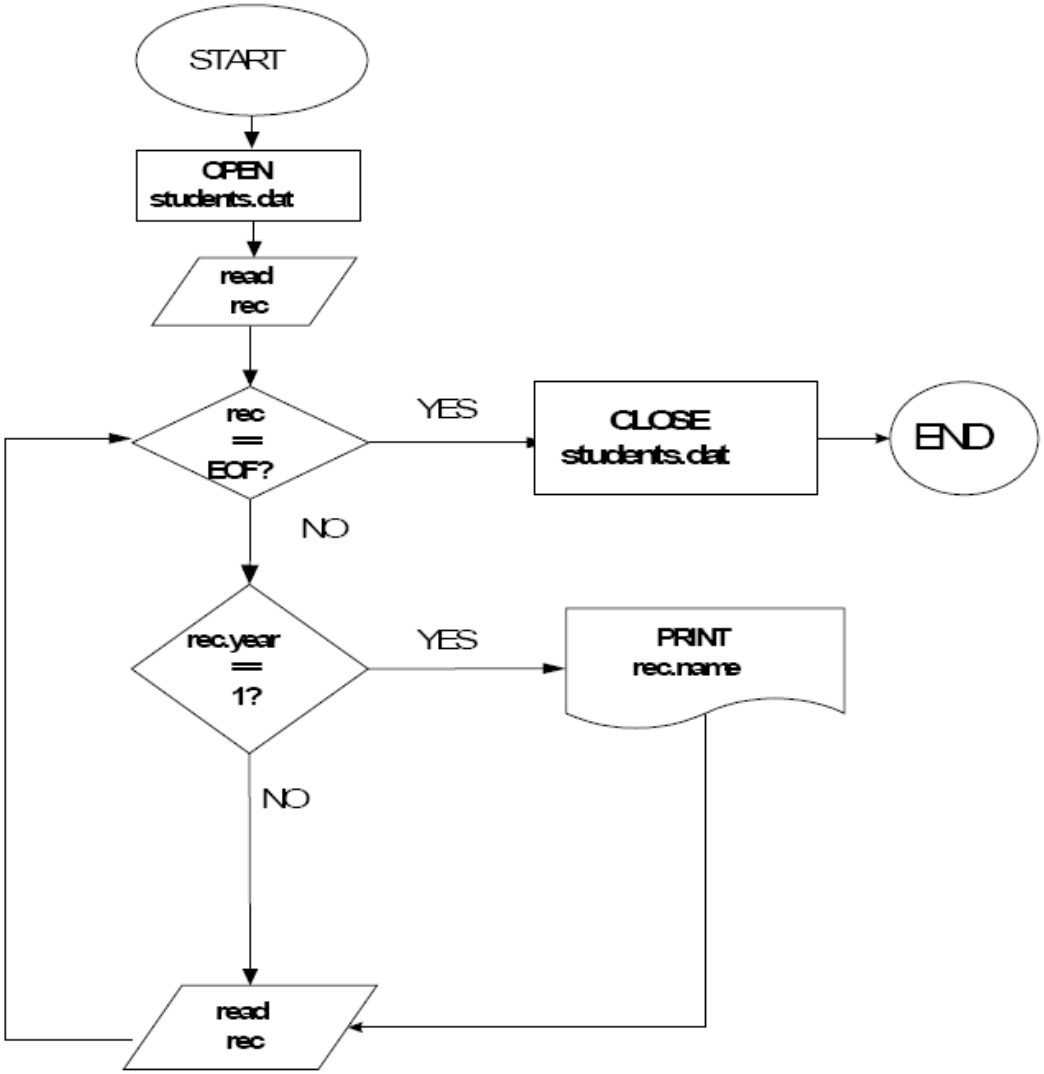
- Δεδομένα Εισόδου:
  - STUDENTS.DAT
- Αποτέλεσμα (Έξοδος):
  - Τα ονόματα των φοιτητών του 1ου έτους.
- Για να επεξεργαστούμε μια εγγραφή θα πρέπει να προσπελασθεί από το δίσκο (DISK) στη μνήμη (RAM)



Παράδειγμα (3/3)

- Λύση:
  1. Άνοιξε το αρχείο.
  2. Διάβασε μια εγγραφή από το δίσκο στη μνήμη.
  3. Αν είναι φοιτητής του πρώτου έτους, τύπωσε το όνομά του.
  4. Αν υπάρχει επόμενη εγγραφή διάβασε την και μετά επανέλαβε το βήμα 3-4.

Διάγραμμα Ροής



Ψευδοκώδικας

```
/* This is a comment */
start                              /* Start program*/
open STUDENT.DAT                   /* Open file */
read rec                           /* Read Record */
while not EOF do                   /* repeat until end */
   if rec.year==1
        then print rec.name
   read rec                        /* Read next record*/
end while
close STUDENT.DAT                  /* Close file */
end                                /* End of program */
```

Έλεγχος

- Επέστρεψε το πρόγραμμα τις αναμενόμενες τιμές?
  - Davis Albert
  - Weller Anthony
  - Costello Barbara
  - Biggert Barney
  - Hastert Baron
- **Ναι!**
  - Για σιγουριά, καλύτερα το πρόγραμμα να ελεγχθεί και με άλλες τιμές εισόδου πριν να χρησιμοποιηθεί.
- **Αν όχι:**
  - Εύρεση και διόρθωση λαθών

Συντήρηση

- Συντήρηση προγράμματος
  - Συγγραφή τεκμηρίωσης
- Αλλαγή συνθηκών προβλήματος
- Οι χρήστες χρειάζονται ακόμη περισσότερα

Ορισμοί

- **Αλγόριθμος:** πεπερασμένη **σειρά ενεργειών**, **αυστηρά καθορισμένων** και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος.
- **Πρόγραμμα:** ονομάζουμε την **αναπαράσταση** ενός ή πολλών αλγορίθμων σε μορφή εκτελέσιμη από τον υπολογιστή
- **Προγραμματισμός:** Το **σύνολο** των **διαδικασιών σύνταξης** ενός υπολογιστικού προγράμματος για την πραγματοποίηση εργασιών ή για την επίλυση ενός δεδομένου προβλήματος.
- **Γλώσσα Προγραμματισμού:** λέγεται μια **τεχνητή γλώσσα** που μπορεί να χρησιμοποιηθεί για τον **έλεγχο μιας μηχανής**, συνήθως ενός υπολογιστή. Οι γλώσσες προγραμματισμού (όπως άλλωστε και οι ανθρώπινες γλώσσες) ορίζονται από ένα σύνολο συντακτικών και εννοιολογικών κανόνων, που ορίζουν τη δομή και το νόημα, αντίστοιχα, των προτάσεων της γλώσσας.
- **Ψευδοκώδικας:** εργαλείο που χρησιμοποιείται στο στάδιο της σχεδίασης ενός προγράμματος για να περιγραφούν τα βήματα του σε σύντομες και **σαφείς** προτάσεις, οι οποίες που υπακούουν σε μια τυποποίηση που πλησιάζει την τυποποίηση μιας γλώσσας προγραμματισμού.
- **Γλώσσα μηχανής:** μια γλώσσα προγραμματισμού που περιλαμβάνει εντολές γραμμένες σε μορφή ακολουθιών bits **άμεσα εκτελέσιμες** από την Κεντρική Μονάδα Επεξεργασίας.
- **Συμβολική γλώσσα (assembly language):** είναι μια χαμηλού επιπέδου γλώσσα προγραμματισμού, σε 1-1 αντιστοιχία με τη γλώσσα μηχανής αλλά γραμμένη με σύμβολα και όρους ευκατανοητους, αντί για 0 και 1.

Κύκλος Ανάπτυξης Προγράμματος

- Περιγραφή του προβλήματος, καθορισμός απαιτήσεων
- Ανάλυση προβλήματος, εύρεση της λύσης
- Σχεδιασμός υλοποίησης της λύσης
  - Ανάπτυξη αλγορίθμου
  - Σχεδιασμός διαγράμματος ροής
  - Δημιουργία ψευδοκώδικα
- Υλοποίηση της λύσης: κωδικοποίηση σε γλώσσα προγραμματισμού
- Έλεγχος, διόρθωση λαθών
- Συντήρηση προγράμματος, τεκμηρίωση
- Βελτιστοποίηση

Περιγραφή του προβλήματος & καθορισμός απαιτήσεων (1/2)

- **Ξεκάθαρη διατύπωση του προβλήματος.**
  - Απομονώνουμε και καταγράφουμε με απλά βήματα τις **πραγματικές συνιστώσες** ενός προβλήματος, τοποθετώντας τις σε **λογική σειρά** μεταξύ τους.
  - Εξαλείφουμε **άσχετες** πληροφορίες
  - Αποσαφηνίζουμε **διφορούμενες** πληροφορίες.
  - Ζητούμε, εάν χρειάζεται, **επιπλέον πληροφορίες** από το άτομο που διατυπώνει το πρόβλημα.
- Αποσαφηνίζουμε τους ***στόχους που επιδιώκουμε να υλοποιήσουμε*** με αναλυτικό τρόπο προκειμένου να καταγραφεί το πλαίσιο **απαιτήσεων** της όλης προσπάθειας.

Περιγραφή του προβλήματος & καθορισμός απαιτήσεων (2/2)

- Πρόβλημα:
  - Το μάθημα ΗΥ150 προσφέρεται από το Τμήμα Επιστήμης Υπολογιστών. Το μάθημα παρακολούθησαν το εαρινό εξάμηνο του 2011, 100 άτομα. Ταξινομήστε τα άτομα αυτά ανάλογα με τη βαθμολογία της τελικής εξέτασης.
- *Επιπλέον Πληροφορίες*:
  - Ποια δεδομένα θα δοθούν (**ονόματα, βαθμοί,** τηλέφωνο,…) και πώς (από **αρχείο**, ή από το χρήστη). Τα αποτελέσματα πως θα παρουσιάζονται? Θα **τυπώνονται** ή θα φυλάσσονται σε αρχείο;
- *Άσχετες Πληροφορίες*:
  - Το μάθημα ΕΠΛ003 προσφέρεται από το Τμήμα Πληροφορικής. Το μάθημα παρακολούθησαν το χειμερινό εξάμηνο του 2011, 100 άτομα.
- *Διφορούμενες Πληροφορίες*:
  - *Ταξινομήστε* (αύξουσα ή **φθίνουσα σειρά**)?

Ανάλυση του προβλήματος & προσδιορισμός της λύσης (1/2)

- Ως μέρος της ανάλυσης απαντούμε στις πιο κάτω ερωτήσεις:
  - ποια είναι τα δεδομένα εισόδου (inputs)
  - ποια είναι τα εξαγώμενα/αποτελέσματα (outputs)
  - ποια είναι η λύση (σχετικές αριθμητικές πράξεις)

Ανάλυση του προβλήματος & προσδιορισμός της λύσης (2/2)

- Σκιαγραφούμε ένα προσχέδιο της επίλυσης του προβλήματος.
- Ελέγχουμε αν η λύση καλύπτει τους στόχους που έχουν τεθεί και αν παράγει τα επιθυμητά δεδομένα εξόδου.
- Διερευνούμε την πιθανότητα ύπαρξης περισσοτέρων λύσεων.
- Επιλέγουμε τη βέλτιστη λύση με βάση τις προδιαγραφές που έχουν τεθεί.
- ΠΡΟΣΟΧΗ: ΛΑΘΟΣ ΑΝΑΛΥΣΗ => ΛΑΘΟΣ ΛΥΣΗ

Αλγόριθμοι

- Γιατί να καταγράψω τον αλγόριθμο;
  - Για ιδίαν χρήση – Δε χρειάζεται να ξανασκεφτείτε το πρόβλημα
  - Ώστε άλλοι να μπορούν να το επιλύσουν, χωρίς να ξέρουν πολλά γύρω από αυτό
    - Δε χρειάζεται να καταλάβουν τις αρχές πίσω από αυτό το πρόβλημα.
    - Απλώς ακολουθούν τις εντολές.
    - Η νοημοσύνη είναι «κωδικοποιημένη στον αλγόριθμο»
  - Για να σας βοηθήσει να καταλάβετε αν επιλύει ορθά το πρόβλημα, να βρείτε αν είναι αποδοτικός

Αλγόριθμοι

- Παραδείγματα αλγορίθμων
  - Οδηγίες πλυντηρίου
  - Οδηγίες για συναρμολόγηση επίπλου
  - Συνταγές
- Στην Πληροφορική
  - Ταξινόμηση Λίστας Αριθμών
  - Εύρεση Μέσου Όρου Μιας Λίστας Αριθμών

Αφαίρεση (Αφαιρετικότητα)-Abstraction

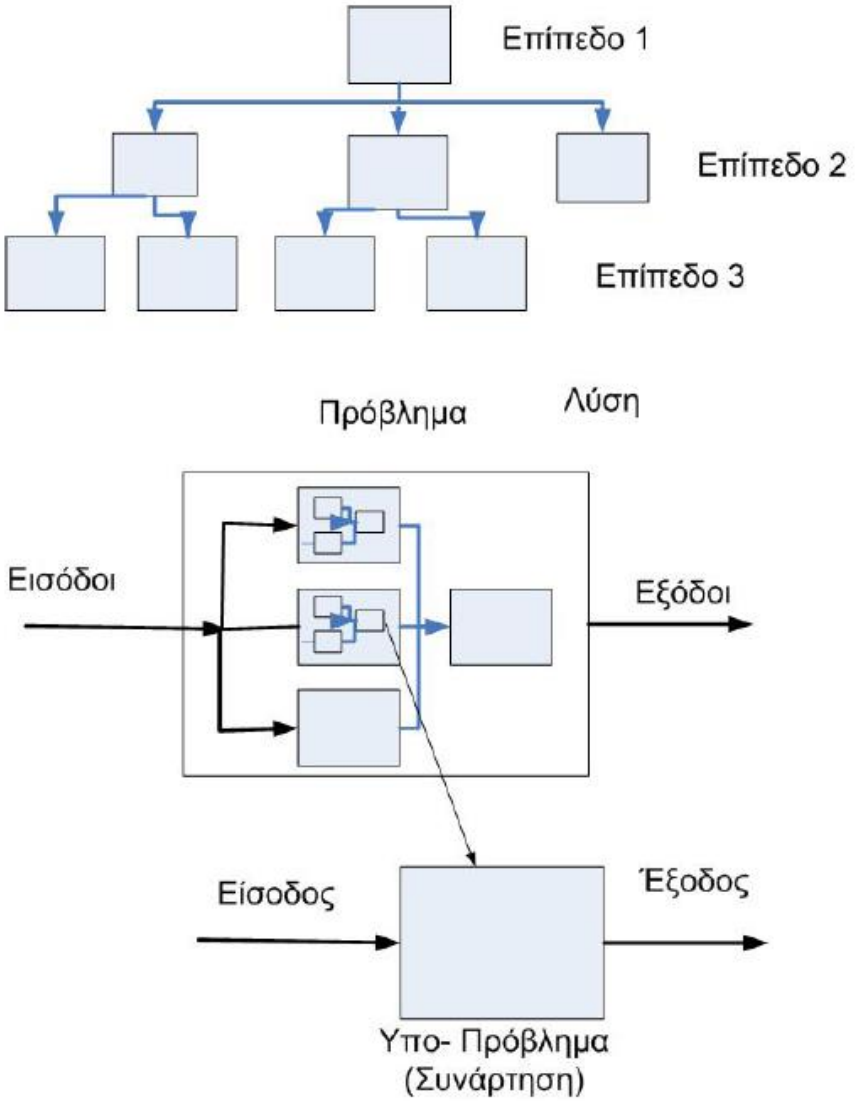
- ***Βασική έννοια στον Προγραμματισμό***
- ***Γνώση του <u>τι</u> γίνεται χωρίς τη γνώση του <u>πως</u> γίνεται***
- Η διαδικασία προγραμματισμού αποτελείται από σχεδιασμό λύσεων σε διάφορα επίπεδα αφαίρεσης

Αφαίρεση

- **Ένας αλγόριθμος εκφράζεται σε διάφορα επίπεδα αφαίρεσης.**
- Είσοδος / Έξοδος: τι είναι το δεδομένο και τι το αποτέλεσμα.
- Η σταδιακή διάσπαση (εκλέπτυνση, ή «πιο λιανά») συνεχίζεται μέχρι έως ότου φτάσουμε σε ατομικά υπο-προβλήματα, δηλαδή προβλήματα που δεν είναι λογικό/δυνατό να διασπαστούν περαιτέρω.
- Στο χαμηλότερο επίπεδο, ο αλγόριθμος διατυπώνει με σαφήνεια την ακριβή διαδικασία παραγωγής της λύσης του προβλήματος

Σχεδιασμός λύσης

- Διάσπαση (εκλέπτυνση) του κυρίως προβλήματος σε υποπροβλήματα (top-down approach, δομημένος, αρθρωτός προγραμματισμός).
- Αφαίρεση. Μας ενδιαφέρει τι γίνεται και όχι το πώς γίνεται.
- Αρχίζοντας από τα χαμηλά επίπεδα αναπτύσσουμε αλγόριθμο για κάθε υποσύστημα.



Παράδειγμα

- Πρόβλημα: Έχουμε μια λίστα με ονόματα. Θέλουμε να βρούμε τους φοιτητές του 1ου έτους

STUDENTS.DAT

Επίθετο	Όνομα	Ηλεκτρονικό Ταχυδρομείο	Έτος	Τηλέφωνο
Davis	Albert	none	1	668-78-9226
Crane	Amory	none	2	689-48-8430
Schakowsky	Anibal	none	3	652-58-7355
Kirk	Anne	rep.kirk@volcano.com	4	623-87-0203
Weller	Anthony	none	1	780-52-0498
Costello	Barbara	none	1	599-98-7962
Biggert	Barney	none	1	058-86-1065
Hastert	Baron	speaker@volcano.com	1	075-45-4923

Ψευδοκώδικας

- Είναι ένα μίγμα αγγλικών και, κοινών σε διάφορες γλώσσες προγραμματισμού, όρων (εντολών), που χρησιμοποιούνται για να εκφράσουμε τη λύση.

```
open STUDENT.DAT              /* Open the file that contains
                                    the student list */
read student_name             /* Read the 1st student name */
while not EOF do              /* Begin the main loop */
   read student_info          /* Read student info */
   if YEAR=1 then
       print student_name     /* conditional structure */
   read student_name          /* Read the next student name */
end while
close STUDENT.DAT             /* Close the input file */
end                           /* End of program */
```

Κώδικας

- Αξιοποιείται η διαδικασία του σχεδιασμού
- Πραγματοποιείται η συγγραφή του προγράμματος σε μια γλώσσα προγραμματισμού.
- Μέσω **μεταγλωττιστή** (compiler) το πρόγραμμα μετατρέπεται σε **γλώσσα μηχανής** η οποία είναι αναγνωρίσιμη από τον υπολογιστή.
- Στο στάδιο αυτό γίνεται ο έλεγχος **συντακτικών** λαθών

Έλεγχος και διόρθωση

- **Σφάλματα σύνταξης (syntax errors):** Σφάλματα που σχετίζονται με το αν χρησιμοποιήσαμε σωστά τη σύνταξη της γλώσσας προγραμματισμού στη διάρκεια της υλοποίησης
- **Λογικά σφάλματα:** σφάλματα που σχετίζονται με το σχεδιασμό της λύσης. Δουλεύει πάντα: λύσε το πρόβλημα με το χέρι για ένα σύνολο δεδομένων και σύγκρινέ το με τις εξόδους του προγράμματος
- **Σφάλματα Run-time:** Σφάλματα κατά τη διάρκεια εκτέλεσης του προγράμματος
  - π.χ. απότομος τερματισμός εκτέλεσης του προγράμματος

Τεκμηρίωση

- Συνοπτική περιγραφή των απαιτήσεων
- Περιγραφή εισόδων, εξόδων, περιορισμών και τύπων
- Ψευδοκώδικας ή διάγραμμα ροής του αλγορίθμου
- Ο ίδιος ο πηγαίος κώδικας (source code)
- Οδηγός για τη χρήση του προγράμματος

Αξιολόγηση

- **Ορθότητα Λύσεων**
  - Αναλυτικές Μέθοδοι – Αποδείξεις
  - Εμπειρικές Μέθοδοι - Δοκιμές
- **Τεκμηρίωση Λύσεων**
  - Σχόλια στον κώδικα
  - Ευκολία κατανόησης
- **Εκτίμηση**
  - **Απόδοσης:** ταχύτητα, ανάγκη σε μνήμη
  - **Ευχρηστίας:** φιλικότητα προς το χρήστη
- **Επεκτασιμότητα / Επαναχρησιμότητα**

Αλγόριθμοι

- Απαιτήσεις - Πρόβλημα – Προδιαγραφές
  - Υπολογίστε το άθροισμα
    1+2+3+…+N
- Σχεδίαση (Υπολογίζω κανονικά το άθροισμα ξεκινώντας από το 1)

```
Σ = 0 //άθροισμα
Για κ=1 … N //μετρητής
    Σ = Σ + κ
```

- Ορθότητα
- Πολυπλοκότητα
  - Υπολογιστικό Κόστος, 2*N -> O(N)
  - Κόστος σε Μνήμη, 2 -> O(1)
- Βελτιστοποίηση: Σ = N*(N+1) / 2
  - Πολυπλοκότητα
    - Υπολογιστικό Κόστος, 3 -> O(1)
    - Κόστος σε Μνήμη, 1 -> O(1)

Παράδειγμα

- Γράψετε ένα πρόγραμμα το οποίο να μετατρέπει μίλια σε χιλιόμετρα.

Βήμα 1. Κατανόηση

- Τι μας ζητά;
  - Τι μίλια;
    - αγγλικά μίλια, ναυτικά μίλια;
- Από πού παίρνουμε τις πληροφορίες;
  - Από το χρήστη, από αρχείο;

Βήμα 2. Ανάλυση

- Δεδομένα (εισόδου): μίλια
- Δεδομένα (εξόδου): χιλιόμετρα
- Άλλα δεδομένα: μαθηματική σχέση
  - 1mile = 1.609Km
- Υπολογισμός (απλή μέθοδος των τριών):
- Παράδειγμα: Πόσα Km είναι 10 μίλια?
  - 1.609*10 = 16.09 χιλιόμετρα

Βήμα 3. Σχεδιασμός λύσης

- Αλγόριθμος

1. Πάρε τα δεδομένα εισόδου
2. Κάνε τη μετατροπή
3. Παρουσίασε το αποτέλεσμα

- 1η Εκλέπτυνση

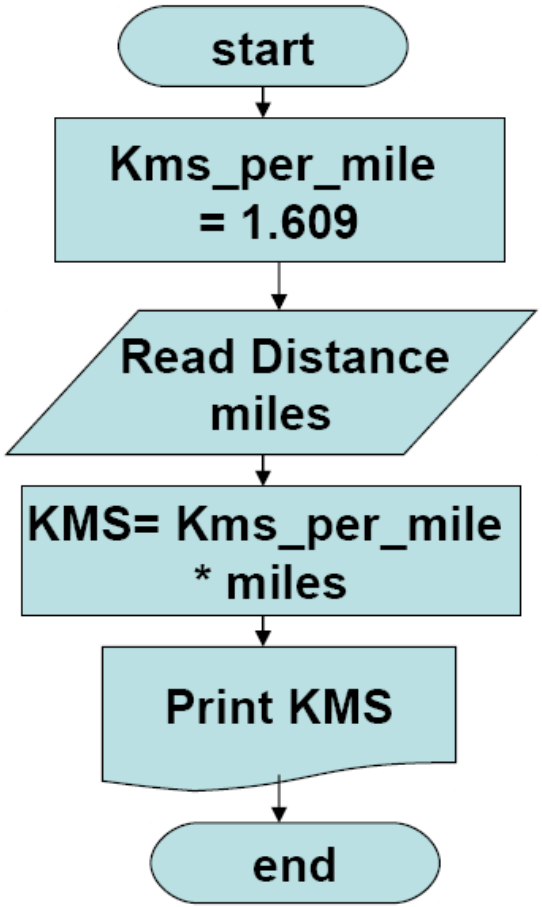
1. Διάβασε τα Μίλια
2. Κάνε τον υπολογισμό: Χλμ=Μίλια * 1.609
3. Δείξε το αποτέλεσμα στην οθόνη

Βήμα 3. Σχεδιασμός λύσης

- 2η Εκλέπτυνση

1. Διάβασε τα Μίλια και αποθήκευσέ τα στη μεταβλητή miles
2. Κάνε τον υπολογισμό: Kms=Miles * 1.609
3. Δείξε το αποτέλεσμα στην οθόνη

- Διάγραμμα Ροής



Ένα πρώτο πρόγραμμα

```
#include <iostream>
using namespace std;

int main()                          // main() is where a C++ program starts
{
    cout << "Hello, world!\n";      // output the 13 characters Hello, world!
                                    // followed by a new line
    return 0;                       // return a value indicating success
}

// quotes delimit a string literal
// NOTE: "smart" quotes "  "  will cause compiler problems.
//       so make sure your quotes are of the style  " "
// \n is a notation for a new line
```

Hello, world!

- Its purpose is to help you get used to your tools
  - Compiler
  - Program development environment
  - Program execution environment
- Type in the program **carefully**
  - After you get it to work, please make a few mistakes to see how the tools respond; for example
    - Forget the header
    - Forget to terminate the string
    - Misspell **return** (e.g. **retrun**)
    - Forget a semicolon
    - Forget **{** or **}**

So what is programming?

- Conventional definitions
  - Telling a **very** fast moron ***exactly*** what to do
  - A plan for solving a problem on a computer
  - Specifying the order of a program execution
    - But modern programs often involve millions of lines of code
    - And manipulation of data is central
- Definition from another domain (academia)
  - A … program is an organized and directed accumulation of resources to accomplish specific … objectives …
    - Good, but no mention of actually doing anything
- The definition we'll use
  - Specifying the structure and behavior of a program, and testing that the program performs its task correctly and with acceptable performance
    - Never forget to check that "it" works
- Software == one or more programs

Programming

- Programming is fundamentally simple
  - Just state what the machine is to do
- So why is programming hard?
  - We want "the machine" to do complex things
    - And computers are nitpicking, unforgiving, dumb beasts
  - The world is more complex than we'd like to believe
    - So we don't always know the implications of what we want
  - "Programming is understanding"
    - When you can program a task, you understand it
    - When you program, you spend significant time trying to understand the task you want to automate
  - Programming is part practical, part theory
    - If you are just practical, you produce non-scalable unmaintainable hacks
    - If you are just theoretical, you produce toys

Μορφοποίηση

- Σχόλια πριν από **συναρτήσεις** και τους ορισμούς μεταβλητών
- Κατάλληλη χρήση tabs, παρενθέσεων και αγκυλών
- Κατάλληλη ονοματολογία συναρτήσεων - μεταβλητών

```
/*Epistrefei ton endiameso ari8mo twn a,b,c*/
int getMedian(int a,int b, int c)
{
        int median; /*Endiamesos*/

        if ((a <= b) && (a >= c)) || ((a >= b) && (a <= c))
        {
            median = a;
        }
        else if ((b <= a) && (b >= c)) || ((b >= a) && (b <= c))
        {
            median = b;
        }
        else
        {
            median = c;
        }

        return median;
}
```

Β Μέρος

Εισαγωγή στη C++

Προτερήματα

- λίγες εντολές
- standard
- φορητή
- ισχυρή
- αρθρωτή
- βάση της C++ και της Java
- γρήγορη

Δομή ενός προγράμματος στη C++

- Οι γλώσσες προγραμματισμού είναι ένα σύνολο από συντακτικούς κανόνες (και βιβλιοθήκες).
- Ένα **πρόγραμμα** σε **C++**, αποτελείται από **εντολές** (οργανωμένα συνήθως μέσα σε συναρτήσεις) και **δεδομένα** (μεταβλητές).

Δομή ενός προγράμματος στη C++

- Οδηγίες στον προεπεξεργαστή
  - ενσωμάτωση βιβλιοθηκών
  - δηλώσεις σταθερών
- Δηλώσεις συναρτήσεων
- Ορισμός *κύριας* συνάρτησης (*main*)
  - δηλώσεις μεταβλητών
  - εντολές (εκφράσεις, κλήσεις συναρτήσεων κτλ)
- Υλοποίηση συναρτήσεων
- Σχόλια

Κύρια συνάρτηση: main()

- **Συνάρτηση:** Σύνολο εντολών που προσδιορίζουν τις υπολογιστικές λειτουργίες που θα εκτελεστούν μιας συγκεκριμένης εργασίας).
- Κάθε C++ πρόγραμμα έχει μία συνάρτηση με το όνομα ***main()*** από όπου **αρχίζει η εκτέλεση του προγράμματος**.
- Σύνταξη:

```
int main ()
{
   σώμα συνάρτησης
}
```

Κύρια συνάρτηση: main()

- Το σώμα μίας συνάρτησης αποτελείται από *δηλώσεις* (declarations) και *εντολές* (executable statements).

```
int main ()
{
   int a;
   a = 5;
   cout <<”Hello ” << a << “ \n”;
   return 0;
}
```

Ροή Έλεγχου (control flow)

- Η ροή ελέγχου στη C είναι ακολουθιακής μορφής - sequential
  - Ξεκινά από την πρώτη εντολή της main()
  - Εντολές εκτελούνται σε σειρά
- Υπάρχουν γλώσσες μή-ακολουθιακές αλλά δε θα μας απασχολήσουν σ’ αυτό το μάθημα.

Σχόλια

- Σύνταξη: ξεκινούν με **/*** και τελειώνουν με ***/**
- Καμιά συνέπεια αλλά πολύ σημαντικό μέρος ενός προγράμματος (τεκμηρίωση)
  - Ο μεταγλωττιστής αγνοεί τα σχόλια

```
/* auto einai ena sxolio*/
// kai auto einai sxolio gia mia grammh
/* alla kai auto
* einai
  ena
* Sxolio pollaplwn grammwn
*/
```

Παράδειγμα

```
/* Filename: 00-my_first_program.c
   Author: X. Zabulis
   Function: Prints the following string of characters:
                              Hello world!
*/
#include <iostream>
using namespace std;

int main()
{
   cout << "Hello world!\n";
   return 0;
}
```

Run

myprogram.cpp

```
> g++ myprogram.cpp
> ls
  a.out
> ./a.out
  Hello World!
```