ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΥΠΟΛΟΓΙΣΤΩΝ

Θεωρία + ΑΠ: Ν. Βασιλάς, Ν. Ζάχαρης

Εργαστήρια: Κ. Κουκουλέτσος, Εμ. Μπρατσόλης, Γ. Τσελίκης, Γ. Μελετίου

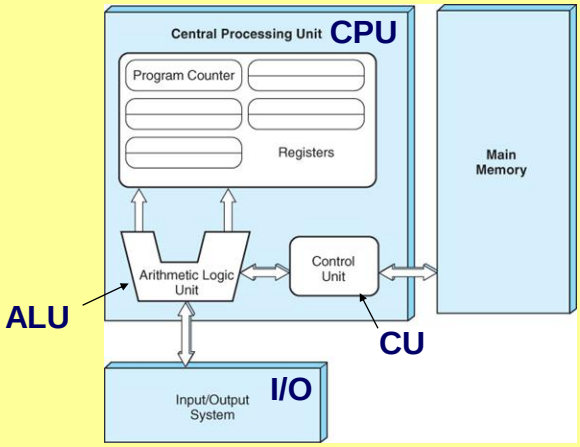
Ο ψηφιακός Η/Υ (hardware)

- Η Von Neumann αρχιτεκτονική

Ο Ούγγρος Μαθηματικός Von Neumann (1903-1957)

Ήδη, από τη δεκαετία του 1940 και ενώ βρισκόταν στις Η.Π.Α. (Princeton University / Institute for Advanced Study), πρότεινε το βασικό μοντέλο (αρχιτεκτονική) Η/Υ που χρησιμοποιείται μέχρι και σήμερα.

Τα μέρη του Η/Υ σύμφωνα με τη Von Neumann αρχιτεκτονική



Ελληνική Ορολογία

CPU = Κεντρική Μονάδα Επεξεργασίας (ΚΜΕ)

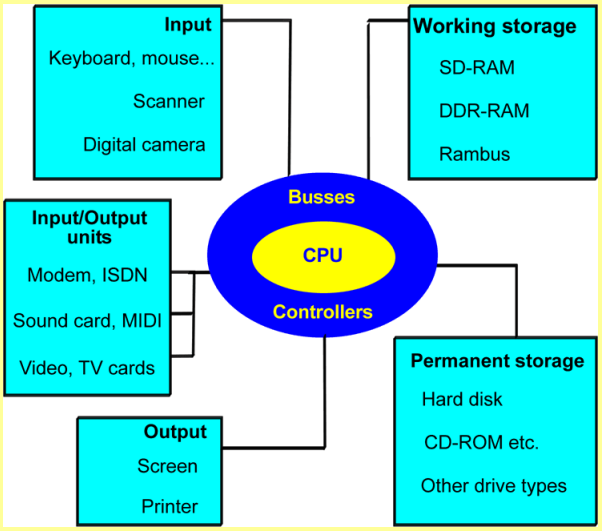
ALU = Μονάδα Αριθμητικών και Λογικών Πράξεων

CU = Μονάδα Ελέγχου

I/O = Είσοδος/Έξοδος

Memory = Μνήμη

Η Von Neumann αρχιτεκτονική σήμερα

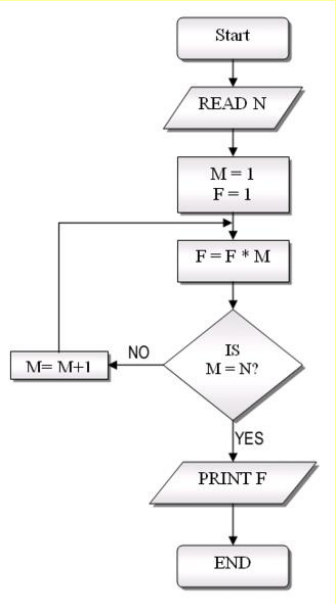


Η εφαρμογή / Το πρόβλημα

- Μαθηματικοί υπολογισμοί (υπολογισμός του Ν!, αντιστροφή πίνακα, ρίζες πολυωνύμων, κ.λπ.)
- Ταξινόμηση συμβολοσειρών, ονομάτων, αριθμών
- Αναζήτηση δεδομένων από βάσεις δεδομένων
- Σχεδιασμός και υλοποίηση πληροφοριακού συστήματος
- Προβλήματα Τεχνητής Νοημοσύνης (σκάκι, αναγνώριση προσώπων, πρόβλεψη, το πρόβλημα του περιοδεύοντος πωλητή, κ.λπ.)
- Τρισδιάστατα γραφικά, εικονική πραγματικότητα

Ο Αλγόριθμος

- Αποτελεσματική μέθοδος που προσδιορίζει τα βήματα που πρέπει να ακολουθήσουμε για την επίλυση προβλημάτων σε Η/Υ
- Κλασικός τρόπος αναπαράστασης αλγορίθμων είναι με διαγράμματα ροής



Υπολογισμός του Ν!

Το Πρόγραμμα (κώδικας)

- Ακολουθία εντολών που εκτελούνται από την ΚΜΕ του Η/Υ
- Επιτρέπει την επικοινωνία με τον Η/Υ
- Είναι σε γλώσσα που «καταλαβαίνει» ο Η/Υ
- Τα πρώτα προγράμματα ήταν γραμμένα σε **γλώσσα μηχανής** (**machine language**) στην οποία οι εντολές και τα δεδομένα είναι σε bits (0/1). Για παράδειγμα, η μείωση κατά 1 της τιμής του καταχωρητή (register) B στη γλώσσα μηχανής του επεξεργαστή Z80 της Zilog δίνεται από την εντολή **00000101**. Η γλώσσα μηχανής είναι μια μή φυσική γλώσσα για τον άνθρωπο

Η Γλώσσα Assembly

- Αντιμετωπίζει εν μέρει τα προβλήματα της γλώσσας μηχανής χρησιμοποιώντας μνημονικά ονόματα στη θέση των δυαδικών εντολών (π.χ. **DEC B** <==> **00000101**)
- Ήταν η βασική γλώσσα προγραμματισμού τη δεκαετία του 1940.
- Ο προγραμματισμός σε γλώσσα assembly απαιτούσε μεγάλη διανοητική προσπάθεια ενώ οι προγραμματιστές που τη χρησιμοποιούσαν ήταν επιρρεπείς σε λάθη

Γλώσσες Υψηλού Επιπέδου (High-Level Languages)

- Αναπτύχθηκαν και χρησιμοποιούνται από τη δεκαετία του 1950
- Επιτρέπουν στον προγραμματιστή να γράφει προγράμματα ανεξάρτητα από τον συγκεκριμένο τύπο Η/Υ
- Οι γλώσσες αυτές μοιάζουν περισσότερο με τις ανθρώπινες γλώσσες σε αντίθεση με την assembly που είναι πιο κοντά στη γλώσσα μηχανής (για τον λόγο αυτό η assembly θεωρείται γλώσσα χαμηλού επιπέδου)
- Βασικό πλεονέκτημα των γλωσσών αυτών: τα προγράμματα σε γλώσσα υψηλού επιπέδου είναι πιο εύκολα στην κατανόηση, στη σύνταξη και στη συντήρηση.
- Προκειμένου να μπορέσουν τελικά να εκτελεστούν στον Η/Υ, τα προγράμματα που έχουν γραφεί σε γλώσσα υψηλού επιπέδου πρέπει να μεταφρασθούν σε γλώσσα μηχανής με χρήση ενός **διερμηνευτή** (interpreter) ή ενός **μεταγλωττιστή** (compiler)
- Σήμερα υπάρχουν δεκάδες διαφορετικών γλωσσών υψηλού επιπέδου. Ενδεικτικά αναφέρουμε σε χρονολογική σειρά τις: *FORTRAN, LISP, ALGOL, COBOL, APL, BASIC, PL/I, Logo, Pascal, C, Smalltalk, Prolog, C++, Ada, Perl, Python, JAVA, Delphi, PHP, XML* και *C#.*

Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ C

- Η C δημιουργήθηκε το 1972 στα Bell Labs από τον Dennis Ritchie καθώς, μαζί με τον Ken Thompson (είχε επινοήσει τη γλώσσα B), σχεδίαζαν το λειτουργικό σύστημα UNIX.

  [Το **λειτουργικό σύστημα** (**operating system**) είναι ένα σύνολο προγραμμάτων για τη διαχείριση των πόρων ενός υπολογιστικού συστήματος].

Ken Thompson →

← Dennis Ritchie

Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ C

- Η C δημιουργήθηκε το 1972 στα Bell Labs από τον Dennis Ritchie καθώς, μαζί με τον Ken Thompson (είχε επινοήσει τη γλώσσα B), σχεδίαζαν το λειτουργικό σύστημα UNIX.

  [Το **λειτουργικό σύστημα** (**operating system**) είναι ένα σύνολο προγραμμάτων για τη διαχείριση των πόρων ενός υπολογιστικού συστήματος].
- Αν και οι περισσότερες γλώσσες στοχεύουν στο να είναι χρήσιμες, συχνά επινοούνται και για άλλους λόγους. Για παράδειγμα, η Pascal σχεδιάσθηκε κυρίως για εκπαιδευτικούς σκοπούς (διδασκαλία αρχών προγραμματισμού) ενώ η BASIC σχεδιάσθηκε ως μια φιλική γλώσσα που μαθαίνεται εύκολα ακόμη και αν δεν υπάρχει εξοικείωση με τους Η/Υ.
- Αντιθέτως, η C σχεδιάσθηκε για να αποτελέσει ένα χρήσιμο εργαλείο για προγραμματιστές.

Η ραγδαία εξάπλωση της C και η καθιέρωσή της ως μιας από τις πιο σημαντικές γλώσσες προγραμματισμού οφείλεται στους εξής λόγους:

- Παρέχει δυνατότητες (π.χ.δομημένου προγραμματισμού) που η υπολογιστική επιστήμη θεωρεί ότι είναι επιθυμητές με αποτέλεσμα πιο αξιόπιστα και κατανοητά προγράμματα.
- Η C είναι μια αποτελεσματική γλώσσα (τα C-προγράμματα έχουν την τάση να είναι πιο συμπαγή και να εκτελούνται πιο γρήγορα).
- Η C είναι μια **μεταφέρσιμη** (**portable**) γλώσσα (ο C-κώδικας που έχει γραφεί σε κάποιο σύστημα μπορεί να χρησιμοποιηθεί αυτούσιος και σε άλλα συστήματα)
- Η C είναι ένα πανίσχυρο και ευέλικτο προγραμματιστικό εργαλείο (π.χ. το 94%, περίπου, του UNIX είναι γραμμένο σε C συμπεριλαμβανομένων και μεταγλωττιστών και διερμηνευτών για άλλες γλώσσες όπως, η FORTRAN, η Pascal, η LISP, η Logo και η BASIC).
- Η C παρέχει δυνατότητες λεπτομερούς ελέγχου της ροής του προγράμματος - κάτι που συνήθως συσχετίζεται με τη γλώσσα assembly - επιτρέποντας βελτιστοποίηση των προγραμμάτων για μέγιστη επίδοση.
- Η C είναι μια φιλική δομημένη γλώσσα που ενθαρρύνει τις καλές προγραμματιστικές συνήθειες.

Ιδιαίτερα χαρακτηριστικά της C

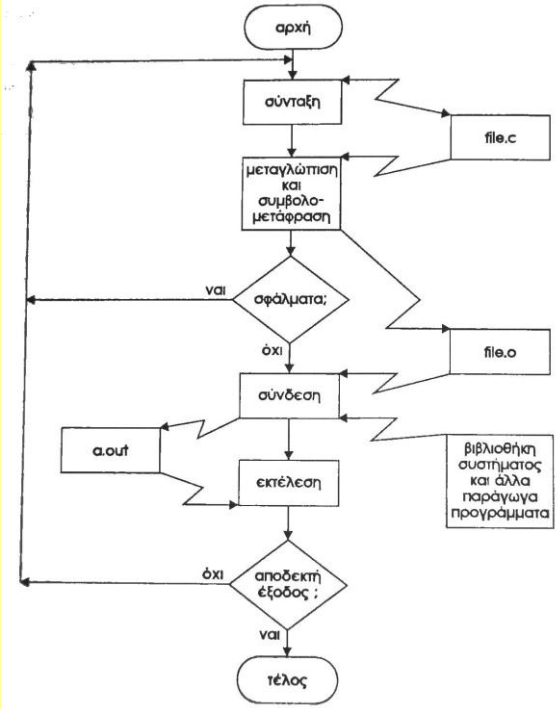
1. Η C αφήνει μεγάλη ελευθερία στον προγραμματιστή. Δεν θέτει πολλούς περιορισμούς και δεν κάνει ελέγχους που τυπικά γίνονται σε άλλες γλώσσες προγραμματισμού (π.χ. η C επιτρέπει την προσπέλαση δεδομένων εκτός των ορίων ενός πίνακα!) δείχνοντας «εμπιστοσύνη» (ότι ξέρει τι κάνει) στον προγραμματιστή.

2. Η C έχει ένα ελάχιστο σύνολο δεσμευμένων λέξεων που την καθιστούν μια μικρή γλώσσα. Όμως, επιτρέπει τη δημιουργία συμπαγούς κώδικα, πολύ μικρότερης έκτασης από άλλες γλώσσες, που μπορεί να είναι εξαιρετικά πολύπλοκος και δυσνόητος. Αυτό την καθιστά, ίσως, «αγαπητή» σε φοιτητές και επαγγελματίες της πληροφορικής αλλά «δύσκολη» και «ακατανόητη» σε φοιτητές από άλλα επιστημονικά πεδία (π.χ. από τις ανθρωπιστικές επιστήμες).

ΣΤΑΔΙΑ ΔΗΜΙΟΥΡΓΙΑΣ ΚΑΙ ΕΚΤΕΛΕΣΗΣ ΕΝΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ ΣΕ C

1. Χρησιμοποιούμε έναν **συντάκτη** (**editor**) για να γράψουμε το πρόγραμμά μας σε ένα **‘.c’ αρχείο**.
2. Υποβάλλουμε το πρόγραμμά μας σε μεταγλώττιση χρησιμοποιώντας τον **μεταγλωττιστή** (**compiler**) της C. Ο μεταγλωττιστής αρχικά θα ελέγξει το πρόγραμμά μας για πιθανά συντακτικά και σημασιολογικά σφάλματα. Αν δεν βρει σφάλματα, θα μεταφράσει το πρόγραμμα σε συμβολική γλώσσα (assembly) και ο συμβολομεταφραστής θα δημιουργήσει ένα **παράγωγο πρόγραμμα** (**object program**) σε **‘.o’ αρχείο**.
3. Στη συνέχεια, γίνεται η σύνδεση του παράγωγου προγράμματός μας με άλλα παράγωγα προγράμματα καθώς και με τις βιβλιοθήκες του συστήματος μέσω του **συνδέτη** (**linker**) και καταλήγουμε στη μορφή ενός **εκτελέσιμου αρχείου** (**a.out**).
4. Εκτελούμε / τρέχουμε (**execute** / **run**) το πρόγραμμά μας πληκτρολογώντας το όνομα του εκτελέσιμου αρχείου.

Διαδικασία παραγωγής εκτελέσιμου προγράμματος



ΕΝΑ ΑΠΛΟ ΠΡΟΓΡΑΜΜΑ

***ΣΗΜΕΙΩΣΗ:** Όλα τα προγράμματα που θα χρησιμοποιηθούν στο μάθημα θα είναι σύμφωνα με το πρότυπο ANSI (American National Standard Institute) της C. Η ANSI C αναπτύχθηκε για να αντιμετωπιστούν προβλήματα ασυμβατότητας.*

Αρχείο **"test.c"**

```
#include <stdio.h>
int main()
{
  printf("Ena poly aplo C-programma.\n");
  return 0;
}
```

- Το πρόγραμμα είναι γνωστό και ως **"πηγαίος κώδικας"** (**source code**) ενώ το αρχείο στο οποίο αποθηκεύεται ονομάζεται **"πηγαίο αρχείο"** (**source file**).
- Για να δημιουργήσουμε εκτελέσιμο αρχείο με χρήση του μεταγλωττιστή cc του UNIX, πληκτρολογούμε:

  `cc test.c`
- Αν δεν υπήρχαν σφάλματα στο πρόγραμμά μας τότε το αποτέλεσμα είναι να δημιουργηθεί ένα εκτελέσιμο αρχείο με όνομα **a.out**. Πληκτρολογώντας τώρα

  `a.out`

  θα εμφανισθεί στην οθόνη του υπολογιστή μας το:

  Ena poly aplo C-programma.

- Παραδείγματα μεταγλώττισης με cc:

  `cc test.c` *ή* `cc test.c foo.c` *(2 αρχεία)*
  παράγει το εκτελέσιμο `a.out`

  `cc -o test test.c`
  αλλάζει το όνομα του εκτελέσιμου σε `test`

  `cc -c test.c`
  δημιουργεί το παράγωγο αρχείο `test.o`. *Δεν καλεί τον συνδέτη για εκτελέσιμο. Χρησιμοποιείται για έλεγχο συντακτικών λαθών ή για σύνδεση με άλλα παράγωγα μέσω της Makefile.*

  `cc -o test test.c -lm`
  σύνδεση με τη μαθηματική βιβλιοθήκη

Επεξηγήσεις

❖ `#include <stdio.h>`

Πριν από κάθε C-πρόγραμμα εμφανίζονται οδηγίες προς τον μεταγλωττιστή μέσω του **προεπεξεργαστή** της C (**C-preprocessor**). Η παραπάνω οδηγία υποχρεώνει τον μεταγλωττιστή της C να διαβάσει και να μεταγλωττίσει το **αρχείο επικεφαλίδας** (**header file**) **`stdio.h`** που έχει σχέση με τη βιβλιοθήκη Εισόδου/Εξόδου στην οποία βρίσκεται η συνάρτηση **`printf()`**.

Το αρχείο **`stdio.h`** (**st**andar**d i**nput **o**utput) περιέχει δηλώσεις συναρτήσεων για εισαγωγή δεδομένων από το πληκτρολόγιο και εμφάνιση δεδομένων στην οθόνη και το συμπεριλαμβάνουμε σε όλα τα προγράμματά μας.

❖
```
int main()
{
  ...
}
```

Ένα C-πρόγραμμα αποτελείται από μια ή περισσότερες **συναρτήσεις** (**functions**). Η βασική συνάρτηση που υπάρχει σε κάθε C-πρόγραμμα είναι η **`main()`**. Μέσα στις παρενθέσεις τοποθετούνται τα **ορίσματα** (**arguments**) της συνάρτησης. Οι παρενθέσεις πρέπει πάντα να υπάρχουν ακόμη και αν δεν υπάρχουν ορίσματα, όπως σ'αυτό το παράδειγμα. Το **σώμα** (**body**) της συνάρτησης, δηλαδή ο κώδικας που αφορά τη συνάρτηση, περικλείεται στη C ανάμεσα από άγκιστρα **{ }**. Η δεσμευμένη λέξη **`int`** υποδηλώνει ότι η **`main()`** πρέπει να επιστρέψει ακέραιο αριθμό στο λειτουργικό σύστημα.

❖ `printf("Ena poly aplo C-programma.\n");`

Η εντολή αυτή είναι στην ουσία μια κλήση της συνάρτησης βιβλιοθήκης `printf()` η οποία, όπως άλλωστε όλες οι δηλώσεις ή εντολές της C, καταλήγει στο σύμβολο `';'`.

Το όρισμα της συνάρτησης αυτής είναι η **συμβολοσειρά** (δηλαδή, η ακολουθία χαρακτήρων) `"Ena poly aplo C-programma.\n"`.

Οι συμβολοσειρές στη C περικλείονται από διπλά εισαγωγικά.

Ο ειδικός χαρακτήρας `'\n'` ανήκει στις λεγόμενες **ακολουθίες διαφυγής** (**escape sequences**) και ονομάζεται **χαρακτήρας νέας γραμμής** (**newline character**).

Το αποτέλεσμα αυτής της εντολής είναι να εμφανισθεί στην οθόνη το

`Ena poly aplo C-programma.`

με τον **δρομέα** (**cursor**) να τοποθετείται στην αρχή της επόμενης γραμμής.

ΑΚΟΛΟΥΘΙΕΣ ΔΙΑΦΥΓΗΣ

`\n`	τοποθέτηση δρομέα στην αρχή νέας γραμμής (newline)
`\t`	χαρακτήρας στηλοθέτη (tab character)
`\b`	χαρακτήρας οπισθοδρόμησης (backspace character)
`\r`	επαναφορά δρομέα στην αρχή της τρέχουσας γραμμής (carriage return)
`\f`	τοποθέτηση δρομέα στην αρχή νέας σελίδας (form feed)
`\\`	ανάποδη κάθετος ή πισωκάθετος ' `\` ' (backslash)
`\'`	απλό εισαγωγικό ' `'` ' (single quote)
`\"`	διπλό εισαγωγικό ' `"` ' (double quote)

ΕΝΑ ΠΙΟ ΣΥΝΘΕΤΟ ΠΡΟΓΡΑΜΜΑ

```
#include <stdio.h>
int main()  /* a simple program */
{
  int num;

  num = 1;
  myprint();
  printf("O arithmos poy brisketai sth ");
  printf("num einai o %d.\n",num);
  return 0;
}

void myprint()
{
  printf("Paradeigma emfanishs arithmou.\n\n");
}
```

Σχόλια

Δήλωση ακέραιας μεταβλητής

Εκχώρηση της τιμής 1 *στην* num

Κλήση δικής μας συνάρτησης

Ορισμός δικής μας συνάρτησης, στο ίδιο αρχείο, μετά την main()

Έξοδος:
```
Paradeigma emfanishs arithmou.

O arithmos poy brisketai sth num einai o 1.
```

ΤΙ ΚΑΝΕΙ ΑΥΤΟ ΤΟ ΠΡΟΓΡΑΜΜΑ?

```
#include <stdio.h>
int main()  /* a simple program */
{
  int num;

  num = 2;
  printf("%d + %d = %d\n",num,num,num+num);
  return 0;
}
```

Έξοδος προγράμματος:

```
2 + 2 = 4
```

ΔΙΟΡΘΩΣΤΕ ΤΟ ΠΑΡΑΚΑΤΩ ΠΡΟΓΡΑΜΜΑ

```
include stdio.h
main{}  /* a simple program /*
(
  int s

  s := 52;
  print(There are s weeks in a year.);
}
```

ΤΟ ΔΙΟΡΘΩΜΕΝΟ ΠΡΟΓΡΑΜΜΑ

```
#include <stdio.h>
int main()  /* a simple program */
{
  int s;

  s = 52;
  printf("There are %d weeks in a year.\n",s);
  return 0;
}
```

Έξοδος προγράμματος:

```
There are 52 weeks in a year.
```