Η γλώσσα C

Γεώργιος Δημητρίου

Προσαρμογή Εισαγωγικού Μαθήματος

από Prof. Stephen A. Edwards

(Columbia University)

Η γλώσσα C

- Σήμερα η πιο διαδεδομένη γλώσσα προγραμματισμού για ενσωματωμένα συστήματα
- “Συμβολική γλώσσα υψηλού επιπέδου”
- Ιδιαίτερα φορητή: μεταγλωττιστές σχεδόν για κάθε επεξεργαστή
- Εύκολη διαδικασία μετάφρασης
- Παραγωγή αποδοτικού κώδικα
- Σχετικά συμπαγής κώδικας

Η ιστορία της C

- Αναπτύχθηκε μεταξύ 1969 και 1973 με το Unix
- Κυρίως από τον Dennis Ritchie
- Σχεδιάστηκε για προγραμματισμό συστημάτων
  - λειτουργικά
  - μεταγλωττιστές
  - βιβλιοθήκες
- Εξέλιξη της B, που με τη σειρά της προήλθε από την BCPL

Η ιστορία της C

- Γράφτηκε όταν οι υπολογιστές ήταν μεγάλες μηχανές
  - Για κάθε υπολογιστή έπρεπε να γράψεις μαζί καινούργια γλώσσα και λειτουργικό σύστημα!
- Η πρώτη μίνι-αρχιτεκτονική (DEC PDP-11, 1974) ήταν πολύ μικρή
  - 24K bytes μνήμης, 12K για το λειτουργικό σύστημα

Η ιστορία της C

- Πολλά χαρακτηριστικά της γλώσσας σχεδιάστηκαν για μείωση της χρησιμοποιούμενης μνήμης
  - Πρόσθιες δηλώσεις απαραίτητες για όλα: μεταβλητές και συναρτήσεις
  - Σχεδιασμένη για να μεταφράζεται σε ένα πέρασμα: ο προγραμματιστής παρέχει όλες τις απαραίτητες πληροφορίες
  - Όχι φωλιασμένες συναρτήσεις

- Για τον ίδιο λόγο, η αρχιτεκτονική PDP-11 είχε διευθυνσιοδότηση σε επίπεδο byte
  - Αυτό στη συνέχεια καθιερώθηκε
  - Άλλα μοντέλα, όπως η διευθυνσιοδότηση λέξης της BCPL, δεν ήταν επαρκή

Το πρόγραμμα «Hello World»

```
#include <stdio.h>

void main()
{
  printf("Hello, world!\n");
}
```

Ο *προεπεξεργαστής* χρησιμοποιείται για κοινοποίηση πληροφορίας μεταξύ αρχείων προγράμματος

- Όχι τόσο εύχρηστος

+ Φθηνή υλοποίηση

+ Πολύ ευέλικτος

Το πρόγραμμα «Hello World»

```
#include <stdio.h>

void main()
{
  printf("Hello, world!\n");
}
```

Ένα πρόγραμμα είναι ένα σύνολο από συναρτήσεις

Η ειδική συνάρτηση *main()* αποτελεί την είσοδο στο πρόγραμμα

Ο τύπος *void* υποδηλώνει συνάρτηση που δεν επιστρέφει αποτέλεσμα (διαδικασία)

Ε/Ε μέσω συνάρτησης βιβλιοθήκης που δεν αποτελεί μέρος της γλώσσας C

Ο αλγόριθμος του Ευκλείδη σε C

```
int gcd(int m, int n)
{
  int r;
  while ( (r = m % n) != 0) {
    m = n;
    n = r;
  }
  return n;
}
```

Η δήλωση συνάρτησης περιλαμβάνει πλήθος και τύπο παραμέτρων, καθώς και τύπο αποτελέσματος.

Οι παράμετροι περνάνε *κατ' αξία*

Ο αλγόριθμος του Ευκλείδη σε C

```
int gcd(int m, int n)
{
  int r;
  while ( (r = m % n) != 0) {
    m = n;
    n = r;
  }
  return n;
}
```

Αυτόματη μεταβλητή

Η δέσμευση μνήμης γίνεται στη *στοίβα* με την είσοδο στη συνάρτηση, απελευθέρωση με την επιστροφή.

Παράμετροι και αυτόματες μεταβλητές διαβάζονται και γράφονται στη στοίβα.

Ο αλγόριθμος του Ευκλείδη σε C

```
int gcd(int m, int n)
{
  int r;
  while ( (r = m % n) != 0) {
    m = n;
    n = r;
  }
  return n;
}
```

Εκφράσεις: Βασικό συστατικό των εντολών της C.

Αριθμητικές και λογικές

Η ανάθεση (=) επιστρέφει αποτέλεσμα, άρα μπορεί να αποτελέσει μέρος έκφρασης!

% → υπόλοιπο

!= → διάφορο

Ο αλγόριθμος του Ευκλείδη σε C

```
int gcd(int m, int n)
{
  int r;
  while ( (r = m % n) != 0) {
    m = n;
    n = r;
  }
  return n;
}
```

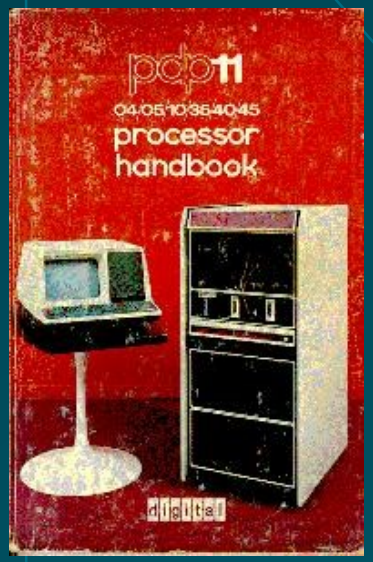
Εντολή ελέγχου ροής υψηλού επιπέδου. Σε χαμηλό επίπεδο γίνεται διακλάδωση.

Αυτό που λέμε «δομημένος προγραμματισμός»

Κάθε συνάρτηση επιστρέφει μοναδική τιμή.

Μετάφραση Ευκλείδη σε PDP-11

```
.globl  _gcd                r0-r7
.text                       PC is r7, SP is r6, FP is r5
_gcd:
    jsr   r5,rsave          save sp in frame pointer r5
L2:mov   4(r5),r1           r1 = n
    sxt   r0                sign extend
    div   6(r5),r0          m / n = r0,r1
    mov   r1,-10(r5)        r = m % n
    jeq   L3
    mov   6(r5),4(r5)       m = n
    mov   -10(r5),6(r5)     n = r
    jbr   L2
L3:mov   6(r5),r0           return n in r0
    jbr   L1
L1:jmp   rretrn             restore sp ptr, return
```



Τι περιλαμβάνει η C

- Τύποι και μεταβλητές
  - Ορισμοί μεταβλητών στη μνήμη
- Εκφράσεις
  - Αριθμητικές και λογικές, καθώς και αναθέσεις, σε ένθετο συμβολισμό
- Εντολές
  - Εντολές έκφρασης
  - Εντολές ελέγχου, συμπεριλαμβανομένων βρόχων και διακλαδώσεων
- Μπλοκ σύνθετης εντολής
  - Ακολουθίες δηλώσεων και εντολών, με πιθανές φωλιασμένες σύνθετες εντολές

Οι τύποι της C

- Βασικοί τύποι: char, int, float, και double
- Προσαρμοσμένοι στους τύπους του επεξεργαστή
  - Φυσική μετάφραση σε γλώσσα μηχανής
  - Για το λόγο αυτό έχουμε και τους τύπους short, long και long long!
  - Πρόβλημα φορητότητας; Ίσως…
- Σύνταξη δηλώσεων: ακολουθίες ονόματος τύπου και προσδιοριστών μαζί με το όνομα της δήλωσης
- Ο συμβολισμός των προσδιοριστών ταιριάζει με το συμβολισμό μιας έκφρασης
- Ένα στοιχείο αναφέρεται με το όνομά του, και μέσω των προσδιοριστών βρίσκουμε το βασικό του τύπο

Παραδείγματα τύπων της C

```
int i;
int *j, k;
unsigned char *ch;
float f[10];
char nC(int, char*);
int a[3][5][10];
int *f1(float);
int (*f2)(void);
```

i: int (= ακέραιος)

j: pointer (= δείκτης) σε int, k: int

ch: δείκτης σε unsigned char

f: πίνακας με 10 στοιχεία float

nC: συνάρτηση 2 παραμέτρων

a: πίνακας 3 πιν. 5 πιν. 10 ακεραίων

f1: συνάρτηση που επιστρέφει int *

f2: δείκτης σε συν. που επιστρέφει int

Δήλωση *typedef*

- Δίνει όνομα σε τύπο, όχι σε μεταβλητή
  - Χρησιμοποιείται σε δηλώσεις σύνθετων – πιθανά αναδρομικών – τύπων

- Πχ. αντί της δήλωσης

```
int (*f2)(void)
```

γράφουμε

```
typedef int f2t(void);
f2t *f2;
```

Πίνακες της C

Filippo Brunelleschi, Ospdale degli Innocenti, Firenze, Italy, 1421

- Πίνακας: ακολουθία ομοειδών αντικειμένων στη μνήμη

```
int a[10];
```

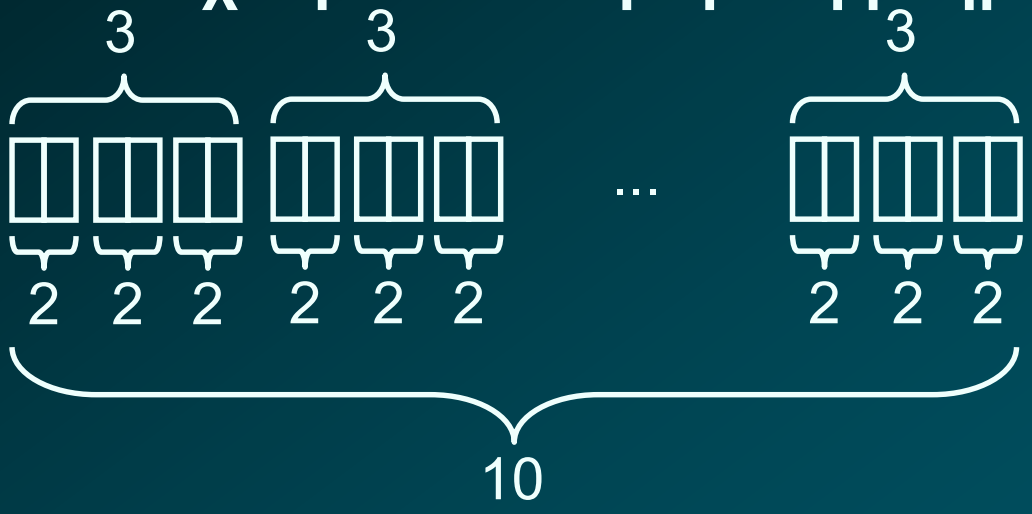
- Με το όνομα a εννοούμε τη διεύθυνση του πρώτου στοιχείου του πίνακα
  - Έτσι *a και a[0] εννοούν το ίδιο πράγμα
- Η διεύθυνση του a δεν αποθηκεύεται στη μνήμη: αν χρειαστούμε τη διεύθυνση αυτή, την υπολογίζουμε

Πολυδιάστατοι πίνακες

- **Πίνακες πινάκων, με τους εσωτερικότερους από τα δεξιά**

  int a[10][3][2];

- **“πίνακας 10 πινάκων 3 πινάκων 2 ακεραίων”**
- **Διαδοχική τοποθέτηση στη μνήμη:**



Seagram Building, Ludwig Mies van der Rohe,1957

Πολυδιάστατοι πίνακες

- Για να περάσουμε έναν πίνακα ως παράμετρο σε συνάρτηση, δε χρειαζόμαστε την πρώτη του διάσταση!

```
int a[10][3][2];
void examine( a[][3][2] ) { … }
```

- Μπορούμε να αναφερθούμε στη θέση του στοιχείου `a[i][j][k]` ως

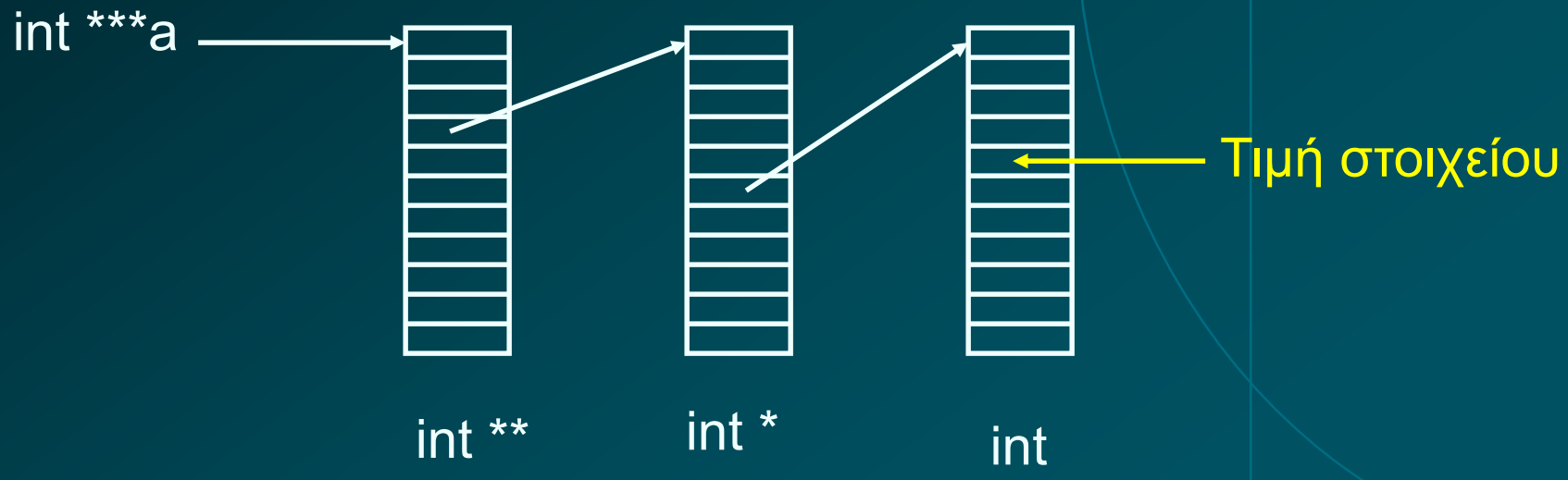
```
a + (i*3 + j)*2 + k
```

Πολυδιάστατοι πίνακες

- Πολυδιάστατοι πίνακες μεταβλητού μεγέθους υλοποιούνται ως πίνακες από δείκτες
- Για τέτοιους πίνακες απαιτείται δέσμευση δυναμικής μνήμης

```
int ***a;
```

- a[3][5][4] είναι το *(*(*(a+3)+5)+4)



Δομή της C (τύπος *struct*)

- Ο τύπος struct ορίζει αντικείμενα με πεδία που έχουν ονόματα

```
struct {
    char *name;
    int x, y;
    int h, w;
} box;
```

- Βλέπουμε κάθε πεδίο με τον τελεστή *dot* (.)

```
box.x = 5;
box.y = 2;
```

Πεδία bit σε δομές struct

- Για συμπίεση δεδομένων στη μνήμη

```
struct {
    unsigned int baud : 5;
    unsigned int div2 : 1;
    unsigned int use_external_clock : 1;
} flags;
```

- Μέγεθος δομής = 7 bits
- Η συμπίεση των πεδίων γίνεται αυτόματα από κώδικα που παράγει ο μεταγλωττιστής
- Όμως όχι αποδοτική υλοποίηση δομής struct
  - Κάθε προσπέλαση πεδίου απαιτεί λειτουργίες bit

Ένωση της C (τύπος *union*)

- Αποθήκευση αντικειμένων διαφορετικού τύπου στον ίδιο χώρο
  - Κάθε φορά βλέπουμε τον τύπο που θέλουμε!

```
union {
    int ival;
    float fval;
    char *sval;
};
```

- Πχ. υλοποίηση πινάκων από διαφορετικά στοιχεία
- Εκφράζει απόλυτα τη φιλοσοφία της C:
  - Ισχυροί μηχανισμοί που μπορούν να είναι επικίνδυνοι!

Κατηγορίες Αποθήκευσης της C

```
#include <stdlib.h>

int global_static;
static int file_static;

void foo(int auto_param)
{
  static int func_static;
  int auto_i, auto_a[10];
  double *auto_d = malloc(sizeof(double)*5);
}
```

Καθολικές, στατική τοποθέτηση

Ορατές μόνο εντός του αρχείου, στατική τοποθέτηση

Ορατές μόνο εντός της συνάρτησης, στατική τοποθέτηση

Κατηγορίες Αποθήκευσης της C

```
#include <stdlib.h>

int global_static;
static int file_static;

void foo(int auto_param)
{
  static int func_static;
  int auto_i, auto_a[10];
  double *auto_d = malloc(sizeof(double)*5);
}
```

Παράμετροι, ορατές μόνο στη συνάρτηση, τοποθέτηση στη στοίβα από τον καλούντα

Αυτόματες, ορατές μόνο στη συνάρτηση, τοποθέτηση στη στοίβα από τη συνάρτηση

Δυναμική εκχώρηση μνήμης

Συναρτήσεις *malloc()* και *free()*

- Ειδικές συναρτήσεις βιβλιοθήκης για τη διαχείριση του *σωρού*

```
int *a;
a = (int *) malloc(sizeof(int) * k);
a[5] = 3;
free(a);
```

- Δυναμική δέσμευση και αποδέσμευση μνήμης οποιουδήποτε μεγέθους σε οποιαδήποτε σειρά
  - Σε αντίθεση με τη στοίβα!
- Τα πιο συνηθισμένα (και δύσκολο να βρεθούν) λάθη

Οι εκφράσεις της C

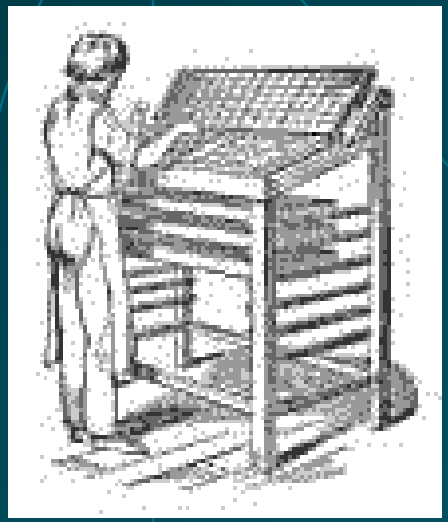
- Παράδειγμα: Κλασικές αλγεβρικές παραστάσεις

```
y = a*x*x + b*x + c;
```

- Πολλές διαφορετικές κατηγορίες εκφράσεων

Κατηγορίες Εκφράσεων της C

- αριθμητικές: + − * / %
- σύγκρισης: == != < <= > >=
- λογικές bit: & | ^ ~
- ολίσθησης: << >>
- λογικές με βραχυκύκλωση: && || !
- συνθήκης: ? :
- ανάθεσης: = += -=
- αύξησης/μείωσης: ++ --
- σειριακής αποτίμησης: ,
- δείκτη/πίνακα/δομής: * -> & . []



Τελεστές Λειτουργιών bit

- and: & or: | xor: ^ not: ~ left shift: << right shift: >>
- Χρήσιμοι για διαχείριση πεδίων bit

```
#define MASK 0x040
if (a & MASK) { … }          /* Check bits */
c |= MASK;                   /* Set bits */
c &= ~MASK;                  /* Clear bits */
d = (a & MASK) >> 4;         /* Select field */
```

Σχόλια της C

Λογικοί Τελεστές με Βραχυκύκλωση

- Ο λογικός έλεγχος με βραχυκύκλωση είναι γρήγορος

  Πχ. η έκφραση

  ```
  if ( a == 3 && b == 4 && c == 5 ) { … }
  ```

  ισοδυναμεί με

  ```
  if (a == 3) { if (b ==4) { if (c == 5) { … } } }
  ```

- Η σειρά αποτίμησης είναι πολύ σημαντική!

  Πχ. για έλεγχο ορίων πίνακα

  ```
  if ( i <= SIZE && a[i] == 0 ) { … }
  ```

Τελεστής συνθήκης

- c = a < b ? a + 1 : b – 1;

- Αποτίμησε την πρώτη έκφραση. Αν αληθής, αποτίμησε τη δεύτερη έκφραση, αλλιώς αποτίμησε την τρίτη έκφραση!

- Στην ουσία υλοποιούμε μια εντολή συνθήκης μέσα σε έκφραση

Πλάγια Αποτελέσματα Εκφράσεων

- Η αποτίμηση μιας έκφρασης μπορεί να έχει *πλάγια αποτελέσματα*

a++	αύξησε το a *μετά την αποτίμηση*
a = 5	άλλαξε την τιμή του a σε 5
a = foo()	και η foo() μπορεί να έχει πλάγια αποτελέσματα

Αριθμητική Δεικτών

- Αναφορά σε κάποιο στοιχείο πίνακα, για παράδειγμα το `*(p+5)` είναι το ίδιο με το `p[5]`

- Αν p και q είναι δείκτες σε κάποιον πίνακα, τότε η έκφραση p−q δίνει το πλήθος στοιχείων μεταξύ των p και q

Εντολές της C

- Εντολή έκφρασης
- Εντολές συνθήκης
  - if (expr) { … } else {…}
  - switch (expr) { case c1: case c2: … }
- Εντολές επανάληψης (βρόχοι)
  - while (expr) { … } — 0 ή περισσότερες επαναλήψεις
  - do … while (expr); — τουλάχιστον 1 επανάληψη
  - for (init ; valid ; next) { … }
- Εντολές άλματος
  - goto label;
  - continue; — πήγαινε στην αρχή του βρόχου
  - break; — τερμάτισε το βρόχο ή το switch
  - return expr; — επέστρεψε από τη συνάρτηση (με την τιμή της έκφρασης expr)

Η Εντολή Switch

- Στην ουσία υλοποιεί σύνθετες διακλαδώσεις

```
switch (expr) {
case 1: …
  break;
case 5:
case 6: …
  break;
default: …
  break;
}
```

```
tmp = expr;
if (tmp == 1) goto L1;
else if (tmp == 5) goto L5;
else if (tmp == 6) goto L6;
else goto Default;
L1: …
  goto Break;
L5:;
L6: …
  goto Break;
Default: …
  goto Break;
Break:
```

Σύνοψη

- Η C μοιάζει με συμβολική γλώσσα υψηλού επιπέδου
- Στηρίζεται σε δομή μνήμης που μοιάζει με πίνακα
- Τα παραπάνω οδηγούν σε πολύ αποδοτική παραγωγή εκτελέσιμου κώδικα!
- Η γλώσσα επιτρέπει σχεδόν τα πάντα
- Άρα είναι πολύ εύκολο να γίνονται λάθη!