
C

Τι μας λείπει ;

Εντολές	`for`
	`while`
	`if και if-else`

Συναρτήσεις εισόδου/εξόδου	`printf()` `scanf()`

Τύποι μεταβλητών	`int`
	`float`
	`char`

Τι καταλαβαίνει ένας υπολογιστής από χαρακτήρες;

Τίποτα.

(ούτε γράμμα)

Δεν καταλαβαίνει C,
Δεν καταλαβαίνει χαρακτήρες,
Τι (επιτέλους) καταλαβαίνει ;

Αριθμούς.

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

0 nul	1 soh	2 stx	3 etx	4 eot	5 enq	6 ack	7 bel
8 bs	9 ht	10 nl	11 vt	12 np	13 cr	14 so	15 si
16 dle	17 dc1	18 dc2	19 dc3	20 dc4	21 nak	22 syn	23 etb
24 can	25 em	26 sub	27 esc	28 fs	29 gs	30 rs	31 us
32 sp	33 !	34 "	35 #	36 $	37 %	38 &	39 '
40 (	41)	42 *	43 +	44 ,	45 -	46 .	47 /
48 0	49 1	50 2	51 3	52 4	53 5	54 6	55 7
56 8	57 9	58 :	59 ;	60 <	61 =	62 >	63 ?
64 @	65 A	66 B	67 C	68 D	69 E	70 F	71 G
72 H	73 I	74 J	75 K	76 L	77 M	78 N	79 O
80 P	81 Q	82 R	83 S	84 T	85 U	86 V	87 W
88 X	89 Y	90 Z	91 [	92 \	93]	94 ^	95 _
96 `	97 a	98 b	99 c	100 d	101 e	102 f	103 g
104 h	105 i	106 j	107 k	108 l	109 m	110 n	111 o
112 p	113 q	114 r	115 s	116 t	117 u	118 v	119 w
120 x	121 y	122 z	123 {	124 |	125 }	126 ~	127 del

ASCII (American Standard Code for Information Interchange)

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

0 nul	1 soh	2 stx	3 etx	4 eot	5 enq	6 ack	7 bel
8 bs	9 ht	10 nl	11 vt	12 np	13 cr	14 so	15 si
16 dle	17 dc1	18 dc2	19 dc3	20 dc4	21 nak	22 syn	23 etb
24 can	25 em	26 sub	27 esc	28 fs	29 gs	30 rs	31 us
32 sp	33 !	34 "	35 #	36 $	37 %	38 &	39 '
40 (	41)	42 *	43 +	44 ,	45 -	46 .	47 /
48 0	49 1	50 2	51 3	52 4	53 5	54 6	55 7
56 8	57 9	58 :	59 ;	60 <	61 =	62 >	63 ?
64 @	65 A	66 B	67 C	68 D	69 E	70 F	71 G
72 H	73 I	74 J	75 K	76 L	77 M	78 N	79 O
80 P	81 Q	82 R	83 S	84 T	85 U	86 V	87 W
88 X	89 Y	90 Z	91 [	92 \	93]	94 ^	95 _
96 `	97 a	98 b	99 c	100 d	101 e	102 f	103 g
104 h	105 i	106 j	107 k	108 l	109 m	110 n	111 o
112 p	113 q	114 r	115 s	116 t	117 u	118 v	119 w
120 x	121 y	122 z	123 {	124 |	125 }	126 ~	127 del

ASCII (American Standard Code for Information Interchange)

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

0 nul	1 soh	2 stx	3 etx	4 eot	5 enq	6 ack	7 bel
8 bs	9 ht	10 nl	11 vt	12 np	13 cr	14 so	15 si
16 dle	17 dc1	18 dc2	19 dc3	20 dc4	21 nak	22 syn	23 etb
24 can	25 em	26 sub	27 esc	28 fs	29 gs	30 rs	31 us
32 sp	33 !	34 "	35 #	36 $	37 %	38 &	39 '
40 (	41)	42 *	43 +	44 ,	45 -	46 .	47 /
48 0	49 1	50 2	51 3	52 4	53 5	54 6	55 7
56 8	57 9	58 :	59 ;	60 <	61 =	62 >	63 ?
64 @	65 A	66 B	67 C	68 D	69 E	70 F	71 G
72 H	73 I	74 J	75 K	76 L	77 M	78 N	79 O
80 P	81 Q	82 R	83 S	84 T	85 U	86 V	87 W
88 X	89 Y	90 Z	91 [	92 \	93]	94 ^	95 _
96 `	97 a	98 b	99 c	100 d	101 e	102 f	103 g
104 h	105 i	106 j	107 k	108 l	109 m	110 n	111 o
112 p	113 q	114 r	115 s	116 t	117 u	118 v	119 w
120 x	121 y	122 z	123 {	124 |	125 }	126 ~	127 del

ASCII (American Standard Code for Information Interchange)

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

Οι κώδικες που καθορίζουν την αντιστοιχία ανάμεσα στους χαρακτήρες μίας (φυσικής) γλώσσας και τους αριθμούς (που χρησιμοποιούνται από τους υπολογιστές για την αντιπροσώπευση των χαρακτήρων) είναι γνωστοί ως (character) encoding sets.

Ο κώδικας ASCII είναι από τους ευρύτερα διαδεδομένους αλλά καλύπτει τους χαρακτήρες που χρησιμοποιούνται από ένα μικρό μόνο σύνολο φυσικών γλωσσών (Αγγλικά, Ινδονησιακά, Σουαχίλη, κλπ). Άλλα encodings περιλαμβάνονται στα διάφορα ISO-8859, όπως το ISO-8859-7 που καλύπτει και την Ελληνική, ή το ISO-8859-14 που καλύπτει την Κελτική, κοκ.

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

Αυτή η ισοδυναμία χαρακτήρων και αριθμών (δηλαδή το encoding) λύνει μόνο ένα μικρό τμήμα του προβλήματος της χρήσης χαρακτήρων από τον υπολογιστή. Για παράδειγμα, η απλή πληκτρολόγηση ενός χαρακτήρα και η εμφάνιση του στο τερματικό σας περιλαμβάνει :

- Ανίχνευση του πλήκτρου που πιέστηκε.
- Αντιστοίχηση του πλήκτρου αυτού (με βάση το encoding και τη γνώση του τύπου του πληκτρολογίου) στον αριθμητικό κωδικό κάποιου χαρακτήρα. Για παράδειγμα, το πλήκτρο v, θα πρέπει στα Αγγλικά (ISO-8859-1) να παράγει το χαρακτήρα «v» (με κωδικό 118), ενώ στα Ελληνικά (ISO-8859-7) θα πρέπει να παράγει το «ω» (με κωδικό 249).

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

- Μετατροπή του κωδικού του χαρακτήρα (και με βάση την εκάστοτε χρησιμοποιούμενη γραμματοσειρά) σε μια γραφική παράσταση της μορφής του χαρακτήρα, η οποία είναι και αυτό που βλέπετε τελικά στην οθόνη του τερματικού σας.

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

– Μετατροπή του κωδικού του χαρακτήρα (και με βάση την εκάστοτε χρησιμοποιούμενη γραμματοσειρά) σε μια γραφική παράσταση της μορφής του χαρακτήρα, η οποία είναι και αυτό που βλέπετε τελικά στην οθόνη του τερματικού σας.

π.χ. Αναπαράσταση του αριθμού 1 σε ένα bitmapped terminal :

```
000000000
000011000          11
000111000         111
000011000          11
000011000          11
000011000          11
001111110        111111
000000000
```

Λύση : αντιστοιχούμε τους χαρακτήρες σε αριθμούς …

- Με αυτό τον τρόπο ερμηνεύεται και η εμφάνιση ακατανόητων συμβόλων (αντί για Ελληνικού κειμένου) σε μερικές περιπτώσεις. Εάν το encoding είναι Ελληνικό αλλά η τρέχουσα γραμματοσειρά είναι Λατινικού τύπου, τότε (για παράδειγμα), αντί για το Ελληνικό «ω», αυτό που θα εμφανιστεί στην οθόνη θα είναι ο χαρακτήρας του Λατινικού encoding (συνήθως ISO-8859-1) που αντιστοιχεί στον κωδικό του «ω», δηλαδή ο χαρακτήρας «ű».

Και με τη C τι σχέση έχουν όλα αυτά ;

Όταν κάνουμε εισαγωγή ή εξαγωγή χαρακτήρων (και ακολουθιών χαρακτήρων) από ένα πρόγραμμα, αυτό που λαμβάνουμε (και στέλνουμε) είναι ο κωδικός των διάφορων χαρακτήρων. Το σε τι αντιστοιχούν (για το χρήστη του προγράμματος) αυτοί οι χαρακτήρες εξαρτάται από το τρέχων encoding και γραμματοσειρά. Για παράδειγμα, οι ακολουθίες χαρακτήρων

hello world !

και

ηελλο ωορλδ !

διαφέρουν μόνο στη χρησιμοποιούμενη γραμματοσειρά (αλλά έχουν ίδιους κωδικούς), …

Και με τη C τι σχέση έχουν όλα αυτά ;

… ενώ οι ακολουθίες χαρακτήρων

hello world !

και

ηελλο ϛορλδ !

διαφέρουν μόνο στο encoding (αλλά μοιράζονται την ίδια γραμματοσειρά) …

Και με τη C τι σχέση έχουν όλα αυτά ;

… ενώ, τέλος, οι ακολουθίες χαρακτήρων

hello world !

και

|ΣLL| ∫|⟩L™ !

διαφέρουν τόσο στο encoding όσο και στη γραμματοσειρά, αλλά παρήχθησαν πληκτρολογώντας την ίδια ακολουθία πλήκτρων που θα απαιτούταν για το «hello world !».

Και που είναι το πρόβλημα ;

Θέλετε να γράψετε ένα πρόγραμμα το οποίο να διαβάζει από την καθιερωμένη είσοδο ένα χαρακτήρα και ανάλογα με το εάν είναι «Y» ή «N» (για Yes ή No, αντίστοιχα) να εκτελεί ή όχι κάποιους υπολογισμούς. Το πρόβλημα είναι ότι εάν μεν το encoding είναι, π.χ. Λατινικό, ο έλεγχος που θα πρέπει να κάνετε θα είναι για τις τιμές 89 («Y») και 78 («N»), ενώ εάν το encoding είναι, π.χ. Ελληνικό, οι τιμές (για τα ίδια σύμβολα) θα είναι 213 και 205.

Και ποια είναι η λύση ;

Η πλέον απλή λύση είναι να αγνοήσουμε το πρόβλημα με την εισαγωγή της κάτωθι σύμβασης :

Για όλα τα προγράμματα που πρόκειται να εξετάσουμε και να παράγουμε, θα υποθέσουμε ότι το encoding είναι ASCII (ή ISO-8859-1), η δε γραμματοσειρά μια οποιαδήποτε γραμματοσειρά συμβατή με αυτό το encoding.

Αυτή η σύμβαση είναι απόλυτα συμβατή με την τεράστια πλειοψηφία των δεδομένων που περιέχονται σε σχεδόν όλες τις βάσεις δεδομένων βιολογικού ενδιαφέροντος.

Πίσω στη C …

```c
#include <stdio.h>

int main()
{
        char c1 = 'A';
        char c2;
        char s1[1000] = "This is a test message";
        char s2[1000];

        printf("%c\n", c1);
        c2 = 66;
        printf("%c %c\n", c1, c2 );

        printf("%s\n", s1);
        s2[0] = 'H';
        s2[1] = 'e';
        s2[2] = 'l';
        s2[3] = 'l';
        s2[4] = 'o';
        s2[5] = 0;
        printf("%s\n", s2 );
        return(0);
}
```

```
A
A B
This is a test message
Hello
```

Δήλωση της μεταβλητής (τύπου χαρακτήρα) **c1** και αρχικοποίηση της με την τιμή που αντιστοιχεί στον κωδικό του γράμματος A :

```
char c1 = 'A';
```

Δήλωση της μεταβλητής (τύπου χαρακτήρα) **c2** :

```
char c2;
```

Δήλωση του πίνακα χαρακτήρων **s1** (1000 χαρακτήρων) και αρχικοποίηση του με τα περιεχόμενα της ακολουθίας χαρακτήρων που ακολουθεί :

```
char s1[1000] = "This is a test message";
```

Δήλωση του πίνακα χαρακτήρων **s2** (1000 χαρακτήρων) :

```
char s2[1000];
```

Εκτύπωση του χαρακτήρα η τιμή του οποίου *(ο αριθμητικός κωδικός του οποίου)* περιέχεται στη μεταβλητή **c1** :

```
printf("%c\n", c1);
```

Ανάθεση στη μεταβλητή (τύπου χαρακτήρα) **c2** της αριθμητικής τιμής 66 (η οποία είναι ο κωδικός του B στο ASCII) :

```
c2 = 66;
```

Εκτύπωση των χαρακτήρων οι τιμές των οποίων *(οι αριθμητικοί κωδικοί των οποίων)* περιέχονται στις μεταβλητές **c1** και **c2** :

```
printf("%c %c\n", c1, c2 );
```

Εκτύπωση της ακολουθίας χαρακτήρων οι τιμές των οποίων *(οι αριθμητικοί κωδικοί των οποίων)* περιέχονται στον πίνακα **s1** :

```
printf("%s\n", s1);
```

Ανάθεση των τιμών που αντιστοιχούν στα γράμματα H, e, l, l και o, στις θέσεις 0, 1, 2, 3 και 4 του πίνακα **s2** :

```
s2[0] = 'H';
s2[1] = 'e';
s2[2] = 'l';
s2[3] = 'l';
s2[4] = 'o';
s2[5] = 0;
```

Αυτό το μηδέν τι κάνει εκεί ;

Αυτό το μηδέν τι κάνει εκεί ;

Σηματοδοτεί την λήξη (το τέλος) της ακολουθίας χαρακτήρων.

Εάν δεν υπήρχε τρόπος να σηματοδοτήσουμε το τέλος μιας ακολουθίας χαρακτήρων, τότε δεν θα μπορούσαμε να αποφασίσουμε εάν φτάσαμε στο τέλος της, και θα συνεχίζαμε να διαβάζουμε (δήθεν) κωδικούς χαρακτήρων επ' άπειρον (ή μέχρι να λάβουμε το μήνυμα : segmentation fault, core dump).

Άρα : στη C όλες οι ακολουθίες χαρακτήρων τελειώνουν με ένα κωδικό (NUL) η αριθμητική τιμή του οποίου είναι ίση με το μηδέν.

```c
#include <stdio.h>

int main()
{
        char c1 = 'A';
        char c2;
        char s1[1000] = "This is a test message";
        char s2[1000];

        printf("%c\n", c1);
        c2 = 66;
        printf("%c %c\n", c1, c2 );

        printf("%s\n", s1);
        s2[0] = 'H';
        s2[1] = 'e';
        s2[2] = 'l';
        s2[3] = 'l';
        s2[4] = 'o';
        s2[5] = 0;
        printf("%s\n", s2 );
        return(0);
}
```

```
A
A B
This is a test message
Hello
```

Τέλος, δυσδιάστατοι πίνακες χαρακτήρων …

```
#include <stdio.h>

int main()
{
        char s[2][50] = { "This is a test message",
                          "This is a second test message"};

        printf("%s\n", s[0]);
        printf("%s\n", s[1]);
        printf("%s %s\n", s[0], s[1]);
        printf("%c %c %c %c\n", s[0][0], s[0][1], s[1][0], s[1][1]);
        return(0);
}
```

```
This is a test message
This is a second test message
This is a test message This is a second test message
T h T h
```

Δήλωση και αρχικοποίηση ενός δυσδιάστατου πίνακα χαρακτήρων (2x50) με το όνομα s[][]. Μπορεί να θεωρηθεί ως το ισοδύναμο δυο γραμμών κειμένου, όπου κάθε γραμμή μπορεί να χωρέσει μέχρι και 49 + 1 (το nul) χαρακτήρες :

```
char s[2][50] = { "This is a test message",
                  "This is a second test message"};
```

Εκτύπωση της ακολουθίας χαρακτήρων που βρίσκεται στην πρώτη γραμμή του πίνακα s[][] :

```
printf("%s\n", s[0]);
```

Εκτύπωση της ακολουθίας χαρακτήρων που βρίσκεται στη δεύτερη γραμμή του πίνακα s[][] :

```
printf("%s\n", s[1]);
```

Εκτύπωση των ακολουθιών χαρακτήρων που βρίσκονται στις γραμμές 0 και 1 του πίνακα s[][] :

```
printf("%s %s\n", s[0], s[1]);
```

Εκτύπωση των χαρακτήρων που βρίσκονται :

- Στην πρώτη θέση της πρώτης γραμμής του πίνακα (s[0][0])
- Στην δεύτερη θέση της πρώτης γραμμής του πίνακα (s[0][1])
- Στην πρώτη θέση της δεύτερης γραμμής του πίνακα (s[1][0])
- Στην δεύτερη θέση της δεύτερης γραμμής του πίνακα (s[1][1])

```
printf("%c %c %c %c\n", s[0][0], s[0][1], s[1][0], s[1][1]);
```

```
#include <stdio.h>

int main()
{
        char s[2][50] = { "This is a test message",
                          "This is a second test message"};

        printf("%s\n", s[0]);
        printf("%s\n", s[1]);
        printf("%s %s\n", s[0], s[1]);
        printf("%c %c %c %c\n", s[0][0], s[0][1], s[1][0], s[1][1]);
        return(0);
}
```

```
This is a test message
This is a second test message
This is a test message This is a second test message
T h T h
```

Παραδείγματα …

```c
#include <stdio.h>

int main()
{
        char c;

        for ( c=32 ; c <= 126 ; c++ )
                printf("%4d  %c\n", c, c );

        return(0);
}
```

```
  32
  33  !
  34  "
  35  #
  36  $
  37  %
  38  &
  39  '
  40  (
  41  )
 .  .  .  .
 .  .  .  .
 124  |
 125  }
 126  ~
```

```c
#include <stdio.h>

int main()
{
        char c;

        for ( c=32 ; c <= 126 ; c++ )
                printf("%4d  %c\n", (int)(c) , c );

        return(0);
}
```

```
  32
  33  !
  34  "
  35  #
  36  $
  37  %
  38  &
  39  '
  40  (
  41  )
 .  .  .  .
 .  .  .  .
 124  |
 125  }
 126  ~
```

32	33 !	34 "	35 #	36 $	37 %	38 &	39 '
40 (	41)	42 *	43 +	44 ,	45 -	46 .	47 /
48 0	49 1	50 2	51 3	52 4	53 5	54 6	55 7
56 8	57 9	58 :	59 ;	60 <	61 =	62 >	63 ?
64 @	65 A	66 B	67 C	68 D	69 E	70 F	71 G
72 H	73 I	74 J	75 K	76 L	77 M	78 N	79 O
80 P	81 Q	82 R	83 S	84 T	85 U	86 V	87 W
88 X	89 Y	90 Z	91 [	92 \	93]	94 ^	95 _
96 `	97 a	98 b	99 c	100 d	101 e	102 f	103 g
104 h	105 i	106 j	107 k	108 l	109 m	110 n	111 o
112 p	113 q	114 r	115 s	116 t	117 u	118 v	119 w
120 x	121 y	122 z	123 {	124 |	125 }	126 ~	

```
 32      33 !    34 "    35 #    36 $    37 %    38 &    39 '
 40 (    41 )    42 *    43 +    44 ,    45 -    46 .    47 /
 48 0    49 1    50 2    51 3    52 4    53 5    54 6    55 7
 56 8    57 9    58 :    59 ;    60 <    61 =    62 >    63 ?
 64 @    65 A    66 B    67 C    68 D    69 E    70 F    71 G
 72 H    73 I    74 J    75 K    76 L    77 M    78 N    79 O
 80 P    81 Q    82 R    83 S    84 T    85 U    86 V    87 W
 88 X    89 Y    90 Z    91 [    92 \    93 ]    94 ^    95 _
 96 `    97 a    98 b    99 c   100 d   101 e   102 f   103 g
104 h   105 i   106 j   107 k   108 l   109 m   110 n   111 o
112 p   113 q   114 r   115 s   116 t   117 u   118 v   119 w
120 x   121 y   122 z   123 {   124 |   125 }   126 ~
```

```
#include <stdio.h>

main()
{
        int     i;
        char    c;

        for ( i=32 ; i <= 126 ; i += 8 )
                {
                        for ( c=i ; c < i+8 && c <= 126 ; c++ )
                                printf("%4d %c  ", c, c );

                        printf("\n");
                                        }
}
```

```
 32      33 !    34 "    35 #    36 $    37 %    38 &    39 '
 40 (    41 )    42 *    43 +    44 ,    45 -    46 .    47 /
 48 0    49 1    50 2    51 3    52 4    53 5    54 6    55 7
 56 8    57 9    58 :    59 ;    60 <    61 =    62 >    63 ?
 64 @    65 A    66 B    67 C    68 D    69 E    70 F    71 G
 72 H    73 I    74 J    75 K    76 L    77 M    78 N    79 O
 80 P    81 Q    82 R    83 S    84 T    85 U    86 V    87 W
 88 X    89 Y    90 Z    91 [    92 \    93 ]    94 ^    95 _
 96 `    97 a    98 b    99 c   100 d   101 e   102 f   103 g
104 h   105 i   106 j   107 k   108 l   109 m   110 n   111 o
112 p   113 q   114 r   115 s   116 t   117 u   118 v   119 w
120 x   121 y   122 z   123 {   124 |   125 }   126 ~
```

```
#include <stdio.h>

main()
{
        int     i;
        char    c;

        for ( i=32 ; i <= 126 ; i += 8 )
                {
                        for ( c=i ; c < i+8 && c <= 126 ; c++ )
                                printf("%4d %c  ", (int)(c), c );

                        printf("\n");
                                        }
}
```

Και πως εισαγάγουμε χαρακτήρες και ακολουθίες χαρακτήρων;

```c
#include <stdio.h>

main()
{
        char    c;

        while ( scanf("%c", &c) == 1 )
                printf("%4d\n", c );
}
```

```
# ./a.out
This is a test
  84
 104
 105
 115
  32
 105
 115
  32
  97
  32
 116
 101
 115
 116
  10
```

```c
#include <stdio.h>

main()
{
        char    c;

        while ( scanf("%c", &c) == 1 )
                printf("%4d\n", (int)( c ) );
}
```

```
# ./a.out
This is a test
  84
 104
 105
 115
  32
 105
 115
  32
  97
  32
 116
 101
 115
 116
  10
```

Αυτός είναι ο αριθμητικός κωδικός που αντιστοιχεί στην αλλαγή γραμμής (το RETURN ή ENTER).

```c
#include <stdio.h>

int main()
{
        char    s[1000];

        while ( scanf("%s", s) == 1 )
                printf("%s\n", s );

        return(0);
}
```

```
# ./a.out
This is a test
This
is
a
test
This is not a test but ...
This
is
not
a
test
but
...
```

```c
#include <stdio.h>

int main()
{
        char    s[1000];

        while ( scanf("%s", s) == 1 )
                printf("%s\n", s );

        return(0);
}
```

```
# ./a.out
This is a test
This
is
a
test
This is not a test but ...
This
is
not
a
test
but
...
```

Η printf() δουλεύει κανονικά, γιατί η scanf() βάζει αυτόματα το μηδέν στο τέλος κάθε ακολουθίας χαρακτήρων.

Το «&» που είναι ;

```
#include <stdio.h>

int main()
{
        char    s[1000];

        while ( scanf("%s", s) == 1 )
                printf("%s\n", s );

        return(0);
}
```

```
# ./a.out
This is a test
This
is
a
test
This is not a test but ...
This
is
not
a
test
but
...
```

Τι μας λείπει ;

Εντολές	`for`
	`while`
	`if και if-else`

Συναρτήσεις εισόδου/εξόδου	`printf()` `scanf()`

Τύποι μεταβλητών	`int`
	`float`
	`char`

Τίποτα.

Εφαρμογές

Εφαρμογή 1η : Υπολογισμός μοριακού βάρους πρωτεΐνης από την πρωτοταγή της δομή.

```
ID   MTTA_THEAQ     STANDARD;      PRT;   421 AA.
AC   P14385;
DT   01-JAN-1990 (Rel. 13, Created)
DT   01-APR-1993 (Rel. 25, Last sequence update)
DT   01-NOV-1997 (Rel. 35, Last annotation update)
DE   MODIFICATION METHYLASE TAQI (EC 2.1.1.72) (ADENINE-SPECIFIC
DE   METHYLTRANSFERASE TAQI) (M.TAQI).
GN   TAQIM.
OS   Thermus aquaticus.
OC   Bacteria; Thermus/Deinococcus group; Thermus group; Thermus.
RP   SEQUENCE FROM N.A.
RC   STRAIN=YT1;
RA   Slatko B.E., Benner J.S., Moran L.S., Jager-Quinton T., Simcox T.G.,
RA   van Cott E.M., Wilson G.G.;
RT   "Cloning, sequencing and expression of the Taq I restriction-
KW   Transferase; Methyltransferase; Restriction system; 3D-structure.
SQ   SEQUENCE   421 AA;  47848 MW;  6F7D2081EB7FCE45 CRC64;
     MGLPPLLSLP SNSAPRSLGR VETPPEVVDF MVSLAEAPRG GRVLEPACAH GPFLRAFREA
     HGTGYRFVGV EIDPKALDLP PWAEGILADF LLWEPGEAFD LILGNPPYGI VGEASKYPIH
     VFKAVKDLYK KAFSTWKGKY NLYGAFLEKA VRLLKPGGVL VFVVPATWLV LEDFALLREF
     LAREGKTSVY YLGEVFPQKK VSAVVIRFQK SGKGLSLWDT QESESGFTPI LWAEYPHWEG
     EIIRFETEET RKLEISGMPL GDLFHIRFAA RSPEFKKHPA VRKEPGPGLV PVLTGRNLKP
     GWVDYEKNHS GLWMPKERAK ELRDFYATPH LVVAHTKGTR VVAAWDERAY PWREEFHLLP
     KEGVRLDPSS LVQWLNSEAM QKHVRTLYRD FVPHLTLRML ERLPVRREYG FHTSPESARN
     F
//
```

Απόσπασμα από την καταχώρηση για την πρωτεΐνη TaqI όπως είναι στη βάση δεδομένων Swiss-Prot

http://www.ebi.ac.uk/swissprot/

MGLPPLLSLP SNSAPRSLGR VETPPEVVDF MVSLAEAPRG GRVLEPACAH
GPFLRAFREA HGTGYRFVGV EIDPKALDLP PWAEGILADF LLWEPGEAFD
LILGNPPYGI VGEASKYPIH VFKAVKDLYK KAFSTWKGKY NLYGAFLEKA
VRLLKPGGVL VFVVPATWLV LEDFALLREF LAREGKTSVY YLGEVFPQKK
VSAVVIRFQK SGKGLSLWDT QESESGFTPI LWAEYPHWEG EIIRFETEET
RKLEISGMPL GDLFHIRFAA RSPEFKKHPA VRKEPGPGLV PVLTGRNLKP
GWVDYEKNHS GLWMPKERAK ELRDFYATPH LVVAHTKGTR VVAAWDERAY
PWREEFHLLP KEGVRLDPSS LVQWLNSEAM QKHVRTLYRD FVPHLTLRML
ERLPVRREYG FHTSPESARN F

Η πρωτοταγής αλληλουχία της πρωτεΐνης με το συμβολισμό του ενός γράμματος.

A,	ALA ,	Alanine	, 71.08
C,	CYS ,	Cysteine	, 103.14
D,	ASP ,	Aspartic Acid	, 115.09
E,	GLU ,	Glutamic Acid	, 129.12
F,	PHE ,	Phenylalanine	, 147.18
G,	GLY ,	Glycine	, 57.05
H,	HIS ,	Histidine	, 137.14
I,	ILE ,	Isoleucine	, 113.16
K,	LYS ,	Lysine	, 128.17
L,	LEU ,	Leucine	, 113.16
M,	MET ,	Methionine	, 131.19
N,	ASN ,	Asparagine	, 114.10
P,	PRO ,	Proline	, 97.12
Q,	GLN ,	Glutamine	, 128.13
R,	ARG ,	Arginine	, 156.19
S,	SER ,	Serine	, 87.08
T,	THR ,	Threonine	, 101.11
V,	VAL ,	Valine	, 99.13
W,	TRP ,	Tryptophan	, 186.21
Y,	TYR ,	Tyrosine	, 163.18

Μοριακό βάρος (της πεπτιδικής μορφής) σε Dalton.

MGLPPLLSLP SNSAPRSLGR VETPPEVVDF MVSLAEAPRG GRVLEPACAH
GPFLRAFREA HGTGYRFVGV EIDPKALDLP PWAEGILADF LLWEPGEAFD
LILGNPPYGI VGEASKYPIH VFKAVKDLYK KAFSTWKGKY NLYGAFLEKA
VRLLKPGGVL VFVVPATWLV LEDFALLREF LAREGKTSVY YLGEVFPQKK
VSAVVIRFQK SGKGLSLWDT QESESGFTPI LWAEYPHWEG EIIRFETEET
RKLEISGMPL GDLFHIRFAA RSPEFKKHPA VRKEPGPGLV PVLTGRNLKP
GWVDYEKNHS GLWMPKERAK ELRDFYATPH LVVAHTKGTR VVAAWDERAY
PWREEFHLLP KEGVRLDPSS LVQWLNSEAM QKHVRTLYRD FVPHLTLRML
ERLPVRREYG FHTSPESARN F

Πως θα μας δοθεί η ακολουθία ;;;;

MGLPPLLSLPSNSAPRSLGRVETPPEVVDFMVSLAEAPRGGRVLEPACAH
GPFLRAFREAHGTGYRFVGVEIDPKALDLPPWAEGILADFLLWEPGEAFD
LILGNPPYGIVGEASKYPIHVFKAVKDLYKKAFSTWKGKYNLYGAFLEKA
VRLLKPGGVLVFVVPATWLVLEDFALLREFLAREGKTSVYYLGEVFPQKK
VSAVVIRFQKSGKGLSLWDTQESESGFTPILWAEYPHWEGEIIRFETEET
RKLEISGMPLGDLFHIRFAARSPEFKKHPAVRKEPGPGLVPVLTGRNLKP
GWVDYEKNHSGLWMPKERAKELRDFYATPHLVVAHTKGTRVVAAWDERAY
PWREEFHLLPKEGVRLDPSSLVQWLNSEAMQKHVRTLYRDFVPHLTLRML
ERLPVRREYGFHTSPESARNF

Θέλουμε το πρόγραμμα μας να μπορεί να υπολογίσει το άθροισμα των μοριακών βαρών όλων των αμινοξέων που του δίδονται στην καθιερωμένη είσοδο (με το συμβολισμό του ενός γράμματος) ασχέτως της μορφής που έχει η είσοδος.

Δηλαδή, τα :

```
MGLPPLLS

M G L P P L L S

M G LPPL
LS
```

κοκ, θα πρέπει όλα να δώσουν το ίδιο (σωστό) αποτέλεσμα. Επιπλέον, θα ήταν χρήσιμο εάν το πρόγραμμα μας θα μπορούσε να ανιχνεύσει περιπτώσεις λάθους (όπως για παράδειγμα, η εισαγωγή ενός χαρακτήρα που δεν αντιστοιχεί σε αμινοξύ, π.χ. το «B»).

Η πλέον εύκολη και προφανής λύση φαίνεται να είναι η εξής :

- Αρχικοποίησε την τιμή του μοριακού βάρους στην τιμή μηδέν.
- Άρχισε να διαβάζεις [με την εντολή scanf(”%s”,…)] ακολουθίες χαρακτήρων από την καθιερωμένη είσοδο μέχρι να ανιχνευθεί τέλος εισόδου (EOF, end-of-file).
- Για κάθε χαρακτήρα (από την πιο πρόσφατη ακολουθία που διαβάσαμε) κάνε τα εξής :
  - ΕΑΝ η αριθμητική τιμή του χαρακτήρα είναι ίση με την αριθμητική τιμή του «Α», τότε αύξησε την τιμή του μοριακού βάρους κατά 71.08 που είναι το μοριακό βάρος της αλανίνης, ΑΛΛΙΩΣ,
  - ΕΑΝ η αριθμητική τιμή του χαρακτήρα είναι ίση με την αριθμητική τιμή του «C», τότε αύξησε την τιμή του μοριακού βάρους κατά 103.14 που είναι το μοριακό βάρος της κυστεΐνης, ΑΛΛΙΩΣ,
  - . . . . . (και για τα είκοσι αμινοξέα)
- Τύπωσε την τελική τιμή του μοριακού βάρους.

```
#include <stdio.h>

int main()
{
    char    seq_segment[10000];
    char    c;
    int     i;
    float   mw;

    mw = 0.0;
    while ( scanf("%s", seq_segment) == 1 )
        {
            for ( i=0 ; i < 10000 ; i++ )
                {
                    if ( seq_segment[i] == 'A' )
                        mw += 71.08;
                    else
                        if ( seq_segment[i] == 'C' )
                            mw += 103.14;
                    else
                    ..............................
                    else
                        if ( seq_segment[i] == 0 )
                            break;
                                    }
                                        }
    printf("The molecular weight is %f\n", mw );
}
```

Εάν είναι αλανίνη ...

Εάν είναι κυστεΐνη ...

Εάν φτάσαμε στο τέλος της αλληλουχίας …

```
#  lude <stdio.h>

int      )
{
    cha     q_segment[10000];
    char
    int
    float   mw;

    mw = 0.0;
    while ( scanf("%s",      egment) == 1
        {
            for ( i=0 ; i <      ;
                {
                    if ( seq_se          == 'A' )
                        mw +=
                    else
                                 eq_segmen       'C' )
                                mw += 103.14;

                                ..........
                    else
                        if ( seq_segment[i] == 0 )
                            break;
                                    }
                                }
    ntf("The molecular weight is %f\n", mw );
```

Εάν είναι αλανίνη ...

Εάν εί   υστεΐνη ...

Εάν φ   με στο τέλος της αλληλουχίας ...

```
#  lude <stdio.h>

int  )
{
     cha  q_segment[10000];
     char
     int
     float   mw;

     mw = 0.0;
     while ( scan
          {
               for
```

Εάν είναι αλανίνη ...

Εάν εί ...οστεΐνη ...

Ένα χάος από τουλάχιστον 22 if-else καθένα από τα οποία θα περιέχει τόσο το αμινοξικό σύμβολο όσο και την αριθμητική τιμή του μοριακού βάρους του αντίστοιχου αμινοξέος.

```
                    if ( seq_segment[i] == 0 )
                         break;
                              }
                                   }
     ntf("The molecular weight is %f\n", mw );
```

Εάν φ ...με στο τέλος της αλληλουχίας ...

Θα χρησιμοποιήσουμε την ισοδυναμία χαρακτήρων και αριθμών προς όφελος μας :

Υποθέστε ότι έχουμε στη διάθεση μας έναν πίνακα τύπου float (ας πούμε με το όνομα weights[]) ο οποίος :
στη θέση 65 (που είναι ο κωδικός του συμβόλου της αλανίνης, A) περιέχει το μοριακό βάρος της αλανίνης (71.08),
στη θέση 67 (που είναι ο κωδικός του συμβόλου της κυστεΐνης, C) περιέχει το μοριακό βάρος της κυστεΐνης(103.14),
κοκ.
ενώ σε όλες τις άλλες θέσεις του περιέχει έναν αρνητικό αριθμό.

Τότε για να βρούμε το μοριακό βάρος της πρωτείνης αρκεί για κάθε αμινοξύ να προσθέτουμε την τιμή που περιέχει ο πίνακας weights[] στη θέση που αντιστοιχεί στον κωδικό του αμινοξέος. Εάν η τιμή που βρίσκουμε από τον weights[] είναι αρνητική, τότε υπάρχει σφάλμα εισόδου.

Αρχικοποίηση του πίνακα ...

```
for ( i = 0 ; i < 127 ; i++ )      /* Για αρχή δώσε αρνητικές   */
    weights[i] = -1.0;             /* τιμές σε όλα τα μορ. βάρη */

weights['A'] = 71.08 ;             /* ... και στη συνέχεια δώσε */
weights['C'] = 103.14;             /* (θετικές) τιμές σε όσα    */
weights['D'] = 115.09;             /* σύμβολα (αμινοξέων)       */
weights['E'] = 129.12;             /* γνωρίζουμε.               */
weights['F'] = 147.18;
weights['G'] = 57.05 ;
weights['H'] = 137.14;
weights['I'] = 113.16;
weights['K'] = 128.17;
weights['L'] = 113.16;
weights['M'] = 131.19;
weights['N'] = 114.10;
weights['P'] = 97.12 ;
weights['Q'] = 128.13;
weights['R'] = 156.19;
weights['S'] = 87.08 ;
weights['T'] = 101.11;
weights['V'] = 99.13 ;
weights['W'] = 186.21;
weights['Y'] = 163.18;
```

Θα χρησιμοποιήσουμε την ισοδυναμία χαρακτήρων και αριθμών προς όφελος μας :

Υποθέστε ότι έχουμε στη διάθεση μας έναν πίνακα τύπου float (ας πούμε με το όνομα weights[]) ο οποίος :
στη θέση 65 (που είναι ο κωδικός του συμβόλου της αλανίνης, A) περιέχει το μοριακό βάρος της αλανίνης (71.08),
στη θέση 67 (που είναι ο κωδικός του συμβόλου της κυστεΐνης, C) περιέχει το μοριακό βάρος της κυστεΐνης(103.14),
κοκ.
ενώ σε όλες τις άλλες θέσεις του περιέχει έναν αρνητικό αριθμό.

Τότε για να βρούμε το μοριακό βάρος της πρωτείνης αρκεί για κάθε αμινοξύ να προσθέτουμε την τιμή που περιέχει ο πίνακας weights[] στη θέση που αντιστοιχεί στον κωδικό του αμινοξέος. Εάν η τιμή που βρίσκουμε από τον weights[] είναι αρνητική, τότε υπάρχει σφάλμα εισόδου.

```c
#include <stdio.h>

int     main()
{
    float   mw;
    float   weights[127];
    char    s[10000];
    int i;

    /*... ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΟΥ ΠΙΝΑΚΑ weights[] ...*/

    mw = 0.0;
    while( scanf("%s", s) != EOF )
        {
            i=0;
            while( s[i] != 0 )
                {
                    if ( weights[ s[i] ] < 0.0 )
                        printf("Unknown character : %c\n", s[i] );
                    else
                        mw += weights[ s[i] ];

                    i++;
                            }
                                }
    printf("The molecular weight is %f\n", mw );
}
```

Εισαγωγή σχολίων (που αγνοούνται από το μεταγλωττιστή) στον πηγαίο κώδικα της C.

```
#include <stdio.h>

int     main()
{
    float   mw;
    float   weights[127];
    char    s[10000];
    int i;

    /*... ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΟΥ ΠΙΝΑΚΑ weights[] ...*/

    mw = 0.0;                                   /* Αρχικοποίηση Μ.Β.       */
    while( scanf("%s", s) != EOF )              /* Για κάθε ακολουθία ... */
        {
            i=0;
            while( s[i] != 0 )                  /* Για κάθε χαρακτήρα ... */
                {
                    if ( weights[ s[i] ] < 0.0 )
                        printf("Unknown character : %c\n", s[i] );
                    else
                        mw += weights[ s[i] ];

                    i++;
                        }
                            }
    printf("The molecular weight is %f\n", mw );
}
```

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
          {
             if ( weights[ s[i] ] < 0.0 )
                printf("Unknown character : %c\n", s[i] );
             else
                mw += weights[ s[i] ];

             i++;
                          }
                                  }
printf("The molecular weight is %f\n", mw );
```

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
          {
             if ( weights[ s[i] ] < 0.0 )
                printf("Unknown character : %c\n", s[i] );
             else
                mw += weights[ s[i] ];

             i++;
                          }
                                  }
printf("The molecular weight is %f\n", mw );
```

Για κάθε ακολουθία χαρακτήρων (s[]) που διαβάζουμε από την είσοδο (μέχρι το τέλος της εισόδου).

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
          {
             if ( weights[ s[i] ] < 0.0 )
                printf("Unknown character : %c\n", s[i] );
             else
                mw += weights[ s[i] ];

             i++;
                          }
                                  }
printf("The molecular weight is %f\n", mw );
```

Για κάθε χαρακτήρα αυτής της αλληλουχίας (μέχρι να φτάσουμε στο μηδέν που δηλώνει το τέλος της αλληλουχίας).

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
          {
             if ( weights[ s[i] ] < 0.0 )
                printf("Unknown character : %c\n", s[i] );
             else
                mw += weights[ s[i] ];

             i++;
                          }
                                  }
printf("The molecular weight is %f\n", mw );
```

Είναι γνωστός αυτός ο χαρακτήρας ; (αντιστοιχεί σε κάποιο αμινοξύ ;).

```
mw = 0.0;
while( scanf("%s", s) != EOF )
   {
      i=0;
      while( s[i] != 0 )
         {
            if ( weights[ s[i] ] < 0.0 )
               printf("Unknown character : %c\n", s[i] );
            else
               mw += weights[ s[i] ];

            i++;
                        }
                                 }
printf("The molecular weight is %f\n", mw );
```

Εάν όχι, σφάλμα εισόδου, ...

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
          {
             if ( weights[ s[i] ] < 0.0 )
                printf("Unknown character : %c\n", s[i] );
             else
                mw += weights[ s[i] ];

             i++;
                          }
                                  }
printf("The molecular weight is %f\n", mw );
```

… ΑΛΛΙΩΣ, αύξησε την τρέχουσα τιμή του μοριακού βάρους κατά τόσο όσο η τιμή που περιέχεται στη θέση του πίνακα weights που αντιστοιχεί στον ASCII κωδικό του χαρακτήρα που ελέγχουμε.

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
         {
            if ( weights[ s[i] ] < 0.0 )
               printf("Unknown character : %c\n", s[i] );
            else
               mw += weights[ s[i] ];

            i++;
                        }
                                  }
printf("The molecular weight is %f\n", mw );
```

Ο επόμενος χαρακτήρας ...

Μετατροπή (μόνο για αυτή την εντολή) του τύπου του s[i] από char σε int.

```
mw = 0.0;
while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
          {
             if ( weights[ (int)( s[i] ) ] < 0.0 )
                printf("Unknown character : %c\n", s[i] );
             else
                mw += weights[ (int)( s[i] ) ];

             i++;
                          }
                                  }
printf("The molecular weight is %f\n", mw );
```

Εφαρμογή 2η : Μετατροπή της αλληλουχίας πρωτεϊνών από το συμβολισμό του ενός γράμματος στο πλήρες αμινοξικό όνομα.

Τι θέλουμε να επιτύχουμε ;

```
# ./a.out
FSDFSGFASD
     1  Phenylalanine
     2  Serine
     3  Aspartic Acid
     4  Phenylalanine
     5  Serine
     6  Glycine
     7  Phenylalanine
     8  Alanine
     9  Serine
    10  Aspartic Acid
```

Πάλι, θέλουμε να μην βάλουμε όρους για τη μορφή της εισόδου, και να είμαστε σε θέση να ανιχνεύσουμε σφάλματα εισόδου.

Ο λάθος τρόπος όπως και πριν, είναι να καταφύγουμε σε μια μακρά σειρά από συνδεδεμένες if-else [εάν ο χαρακτήρας είναι «A», τύπωσε «Alanine», αλλιώς, …].

Ένας δεύτερος τρόπος (κάπως καλύτερος) είναι ανάλογος της μεθόδου που χρησιμοποιήσαμε στο προηγούμενο παράδειγμα : Όρισε έναν δυσδιάστατο πίνακα χαρακτήρων 127 γραμμών με κάθε γραμμή να έχει τόσους χαρακτήρες ώστε να χωράει το πλήρες όνομα του αμινοξέος (ας πούμε 127 επί 15). Τοποθέτησε τα πλήρη ονόματα των αμινοξέων στις θέσεις του πίνακα που αντιστοιχούν στους κωδικούς του συμβολισμού ενός γράμματος (π.χ. το «Alanine» στη γραμμή 65 του πίνακα). Για τις γραμμές που δεν αντιστοιχούν σε κάποιο αμινοξύ, τοποθέτησε ένα χαρακτηριστικό αριθμό (κωδικό) στην πρώτη θέση του ονόματος. Αυτός ο τρόπος καταναλώνει άδικα πολύ μνήμη.

Ο τρόπος που θα εφαρμόσουμε δεν σπαταλάει χώρο μνήμης αλλά απαιτεί την εισαγωγή ενός επιπλέον πίνακα ακεραίων. Τα βασικά σημεία αυτού του τρόπου είναι :

- Όρισε ένα πίνακα χαρακτήρων (20 επί 15) στον οποίο θα αποθηκεύσουμε τα πλήρη ονόματα των 20 αμινοξέων

```
char    names[20][15] = {
      "Alanine       ", "Cysteine      ", "Aspartic Acid ",
      "Glutamic Acid ", "Phenylalanine ", "Glycine       ",
      "Histidine     ", "Isoleucine    ", "Lysine        ",
      "Leucine       ", "Methionine    ", "Asparagine    ",
      "Proline       ", "Glutamine     ", "Arginine      ",
      "Serine        ", "Threonine     ", "Valine        ",
      "Tryptophan    ", "Tyrosine      "   };
```

Ο τρόπος που θα εφαρμόσουμε δεν σπαταλάει χώρο μνήμης αλλά απαιτεί την εισαγωγή ενός επιπλέον πίνακα ακεραίων. Τα βασικά σημεία αυτού του τρόπου είναι :

- Όρισε ένα πίνακα χαρακτήρων (20 επί 15) στον οποίο θα αποθηκεύσουμε τα πλήρη ονόματα των 20 αμινοξέων

```
char    names[20][15] = {
      "Alanine       ", "Cysteine      ", "Aspartic Acid ",
      "Glutamic Acid ", "Phenylalanine ", "Glycine       ",
      "Histidine     ", "Isoleucine    ", "Lysine        ",
      "Leucine       ", "Methionine    ", "Asparagine    ",
      "Proline       ", "Glutamine     ", "Arginine      ",
      "Serine        ", "Threonine     ", "Valine        ",
      "Tryptophan    ", "Tyrosine      "   };
```

Το πρόβλημα είναι πως θα αντιστοιχίσουμε τον κωδικό του π.χ. «C» (από την είσοδο) στην θέση 1 του πίνακα names[][].

- Όρισε ένα πίνακα ακεραίων 127 θέσεων και σε όσες θέσεις του πίνακα αντιστοιχούν σε αμινοξύ, βάλε τη θέση του ονόματος του αμινοξέος στον πίνακα names[][] :

```
for ( i = 0 ; i < 127 ; i++ )
        pointers[i] = -1;
pointers['A'] = 0;
pointers['C'] = 1;
pointers['D'] = 2;
pointers['E'] = 3;
pointers['F'] = 4;
pointers['G'] = 5;
pointers['H'] = 6;
pointers['I'] = 7;
pointers['K'] = 8;
pointers['L'] = 9;
pointers['M'] = 10;
pointers['N'] = 11;
pointers['P'] = 12;
pointers['Q'] = 13;
pointers['R'] = 14;
pointers['S'] = 15;
pointers['T'] = 16;
pointers['V'] = 17;
pointers['W'] = 18;
pointers['Y'] = 19;
```

```c
#include <stdio.h>

int   main()
{
  char  s[10000];
  int i;
  int total;
  int pointers[127];
  char    names[20][15] = {  /* Αρχικοποιήσεις πινάκων
  . . . . . . . . . . . . . . . . . . . . . . . .  .. . */
  total = 1;
  while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
        {
          if ( pointers[ (int)(s[i]) ] < 0 )
            printf("Unknown character : %c\n", s[i] );
          else
            {
            printf("%6d  %s\n", total, names[ pointers[(int)(s[i])]]);
            total++;
            }
          i++;
              }
                }
  return(0);
}
```

```
#include <stdio.h>

int   main()
{
  char  s[10000];
  int i;
  int total;
  int pointers[127];
  char    names[20][15] = {  /* Αρχικοποιήσεις πινάκων
  . . . . . . . . . . . . . . . . . . . . . . . .  .. . */
  total = 1;
  while( scanf("%s", s) != EOF )   /* Για κάθε αλληλουχία χαρακτήρων */
    {
      i=0;
      while( s[i] != 0 )      /* Για κάθε χαρακτήρα της αλληλουχίας */
        {
          if ( pointers[ (int)(s[i]) ] < 0 )     /* Είναι γνωστός ? */
            printf("Unknown character : %c\n", s[i] );
          else
            {      /* εάν ναι, τύπωσε το πλήρες όνομα του αμινοξέως */
            printf("%6d  %s\n", total, names[ pointers[(int)(s[i])]]);
            total++;
            }
          i++;
              }
                }

  return(0);
}
```

Η ακεραίου τύπου μεταβλητή total μετράει το συνολικό αριθμό αμινοξέων που έχουν επιτυχώς εισαχθεί.

```
#include <stdio.h>

int   main()
{
  char  s[10000];
  int i;
  int total;
  int pointers[127];
  char    names[20][15] = {  /* Αρχικοποιήσεις πινάκων
  . . . . . . . . . . . . . . . . . . . . . . . .   .. . */
  total = 1;
  while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
        {
          if ( pointers[ (int)(s[i]) ] < 0 )
            printf("Unknown character : %c\n", s[i] );
          else
            {
            printf("%6d  %s\n", total, names[ pointers[(int)(s[i])]]);
            total++;
            }
          i++;
              }
                }
  return(0);
}
```

```
#include <stdio.h>

int   main()
{
  char  s[10000];
  int i;
  int total;
  int pointers[127];
  char    names[20][15] = {  /* Αρχικοποιήσεις πινάκων
  . . . . . . . . . . . . . . . . . . . . . . . .  .. . */
  total = 1;
  while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
        {
          if ( pointers[ (int)(s[i]) ] < 0 )
            printf("Unknown character : %c\n", s[i] );
          else
            {
            printf("%6d  %s\n", total, names[ pointers[(int)(s[i])]]);
            total++;
            }
          i++;
              }
                }
  return(0);
}
```

Ο πίνακας χαρακτήρων s[] περιέχει την πλέον πρόσφατα εισαχθείσα αλληλουχία χαρακτήρων.

```
#include <stdio.h>

int   main()
{
  char  s[10000];
  int i;
  int total;
  int pointers[127];
  char    names[20][15] = {  /* Αρχικοποιήσεις πινάκων
  . . . . . . . . . . . . . . . . . . . . . . . .  .. . */
  total = 1;
  while( scanf("%s", s) != EOF )
    {
      i=0;
      while( s[i] != 0 )
        {
          if ( pointers[ (int)(s[i]) ] < 0 )
            printf("Unknown character : %c\n", s[i] );
          else
            {
            printf("%6d  %s\n", total, names[ pointers[(int)(s[i])]]);
            total++;
            }
          i++;
              }
                }
  return(0);
}
```

Η ακεραίου τύπου μεταβλητή i είναι δείκτης στους χαρακτήρες που περιέχονται στην πιο πρόσφατη ακολουθία χαρακτήρων που διαβάσαμε (περιέχεται στο s[]).

Εάν στη θέση του πίνακα pointers[] που αντιστοιχεί στον κωδικό του γράμματος s[i] υπάρχει αρνητικός αριθμός, τότε υπάρχει σφάλμα εισόδου, ΑΛΛΙΩΣ, …

```
if ( pointers[ (int)(s[i]) ] < 0 )
     printf("Unknown character : %c\n", s[i] );
else
  {
    printf("%6d  %s\n", total, names[ pointers[(int)(s[i])] ] );
    total++;
            }
```

```
if ( pointers[ (int)(s[i]) ] < 0 )
     printf("Unknown character : %c\n", s[i] );
else
  {
    printf("%6d  %s\n", total, names[ pointers[(int)(s[i])] ] );
    total++;
            }
```

Τύπωσε την ακολουθία χαρακτήρων του πίνακα names[][], που βρίσκεται στη θέση …

```
if ( pointers[ (int)(s[i]) ] < 0 )
     printf("Unknown character : %c\n", s[i] );
else
  {
    printf("%6d  %s\n", total, names[ pointers[(int)(s[i])] ] );
    total++;
            }
```

Τύπωσε την ακολουθία χαρακτήρων του πίνακα names[][], που βρίσκεται στη θέση … που ορίζεται από τα περιεχόμενα του πίνακα pointers[] …

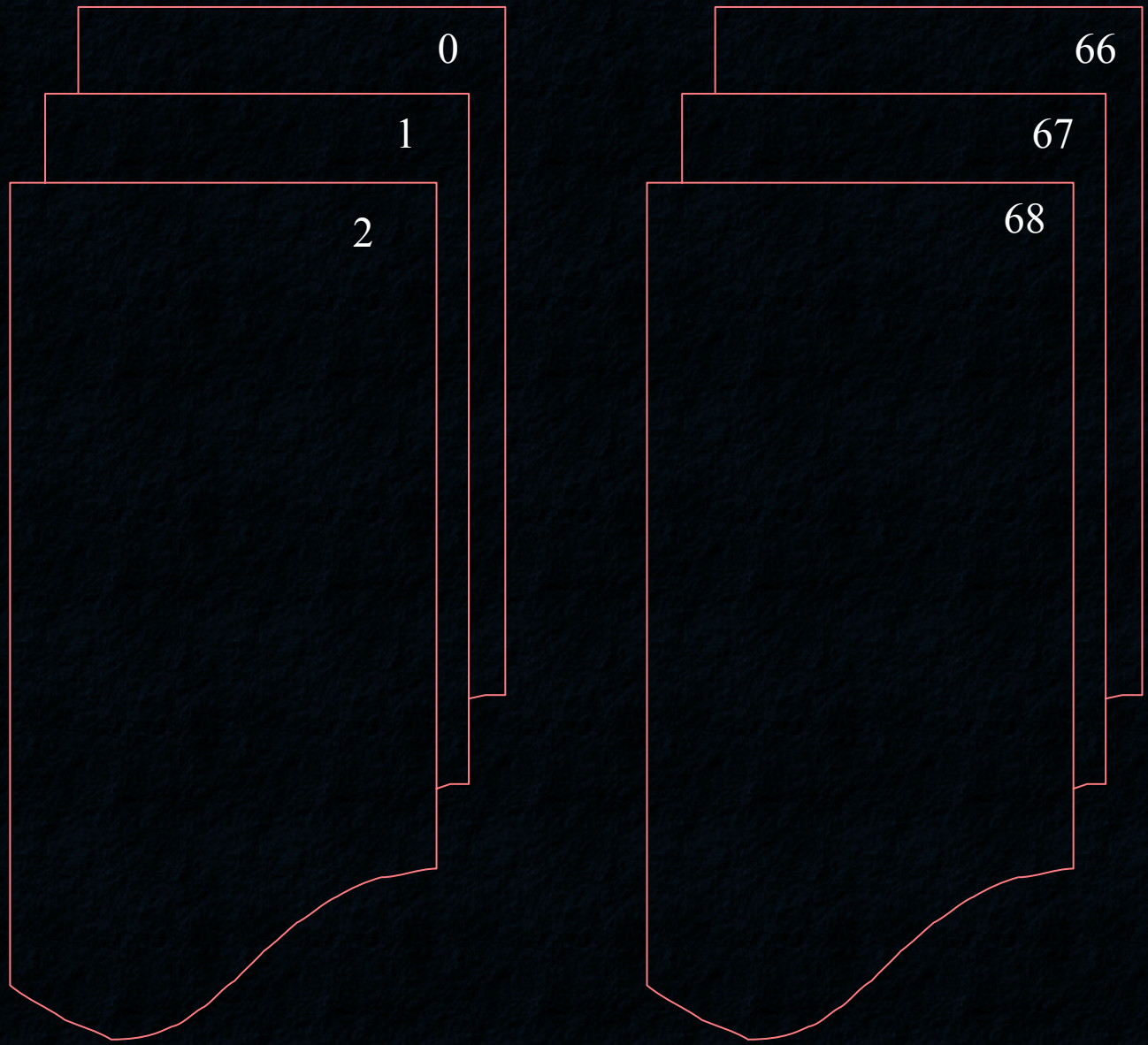
```
if ( pointers[ (int)(s[i]) ] < 0 )
     printf("Unknown character : %c\n", s[i] );
else
  {
    printf("%6d  %s\n", total, names[ pointers[(int)(s[i])] ] );
    total++;
            }
```

Τύπωσε την ακολουθία χαρακτήρων του πίνακα names[][], που βρίσκεται στη θέση … που ορίζεται από τα περιεχόμενα του πίνακα pointers[] … στη θέση του κωδικού του χαρακτήρα που εξετάζουμε s[i]

Πίνακας names[]

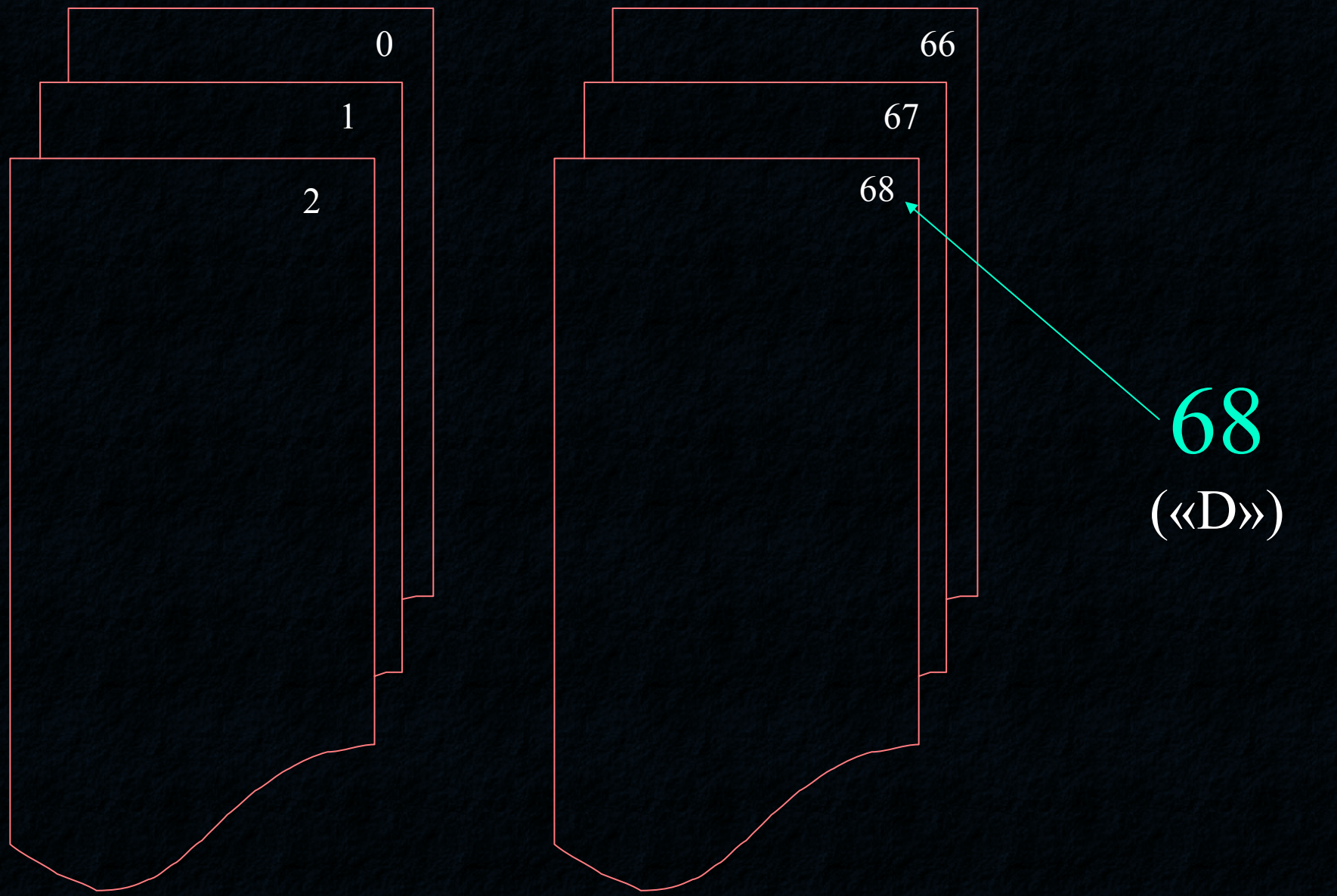
Πίνακας pointers[]

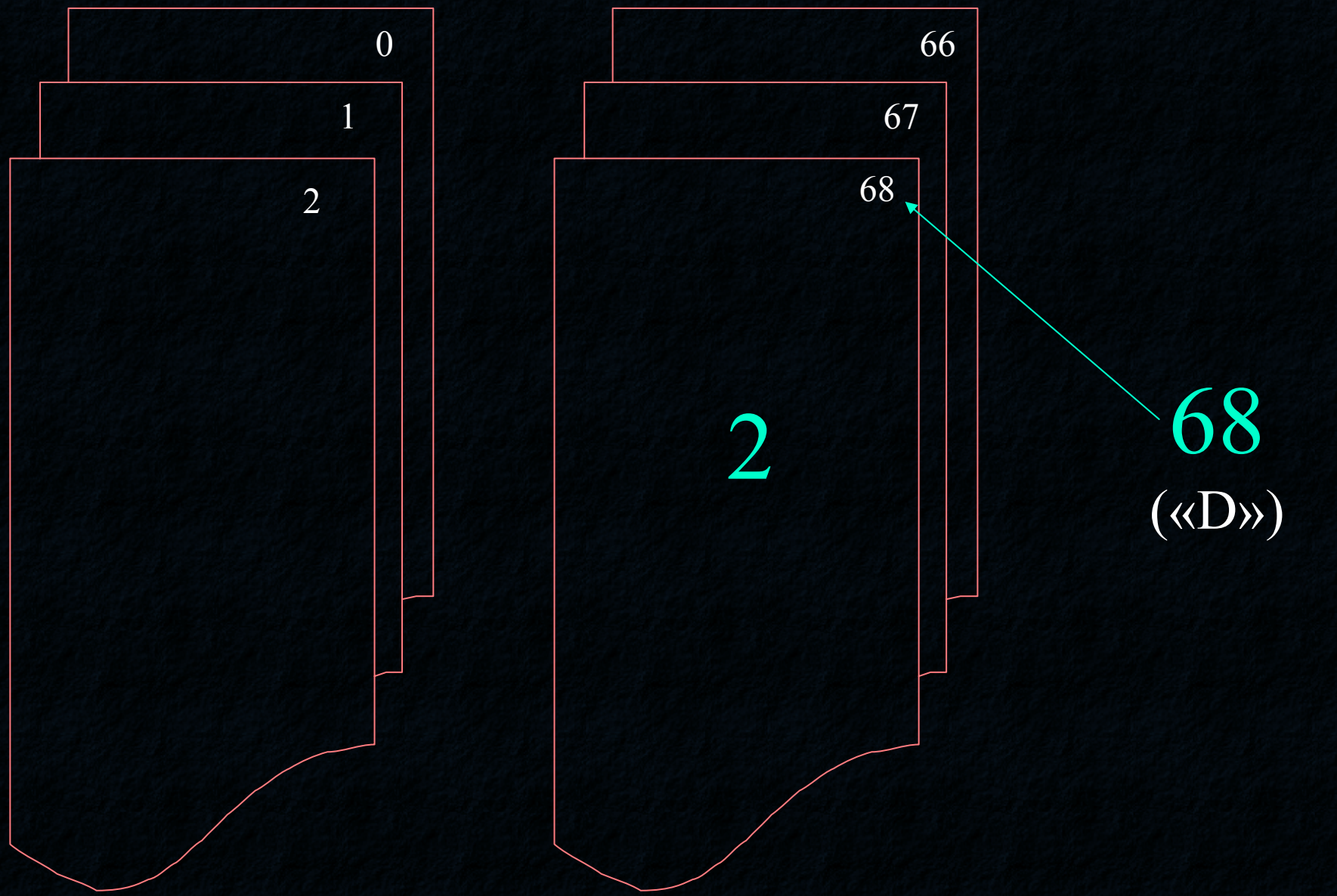
Είσοδος

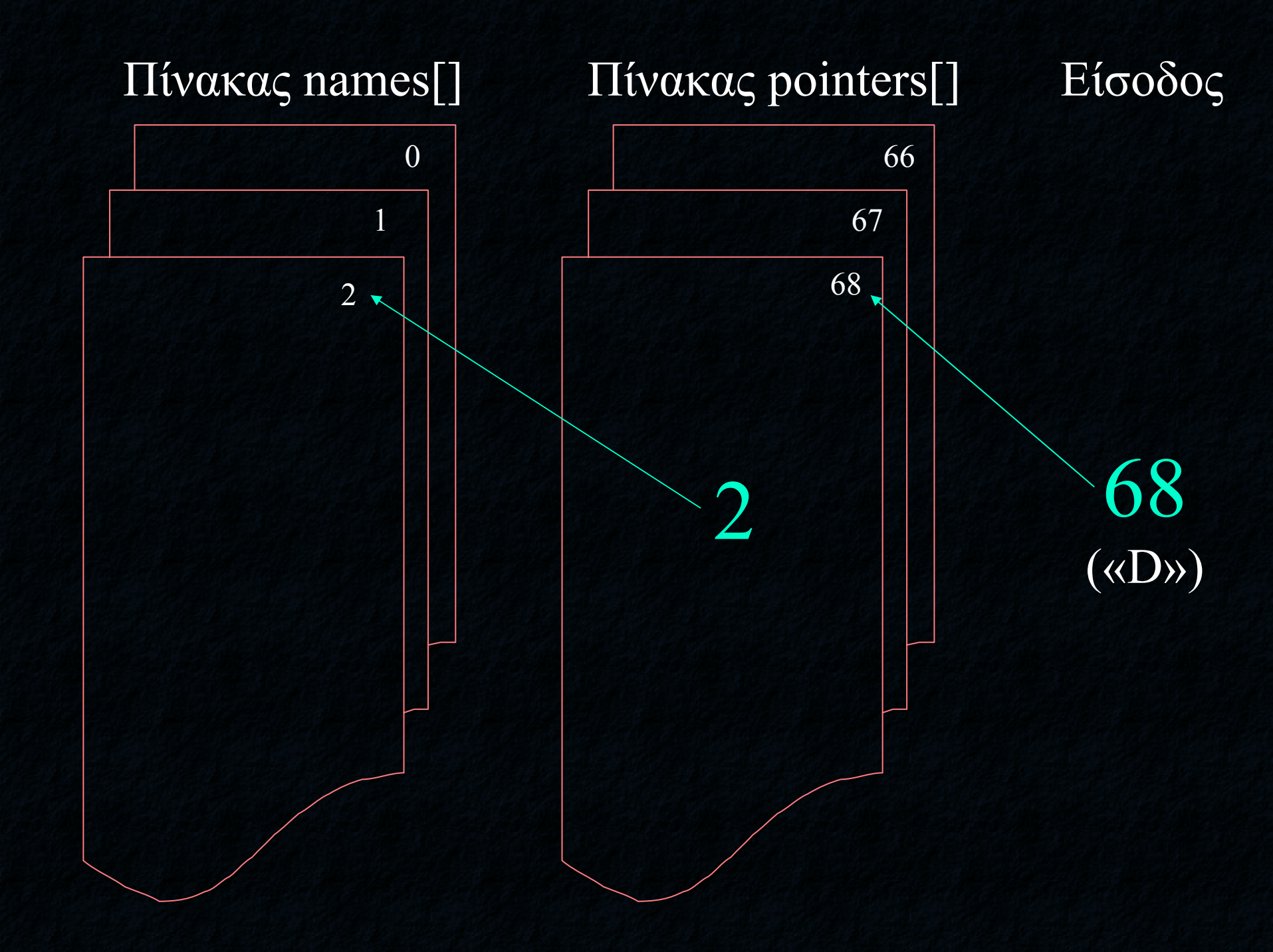


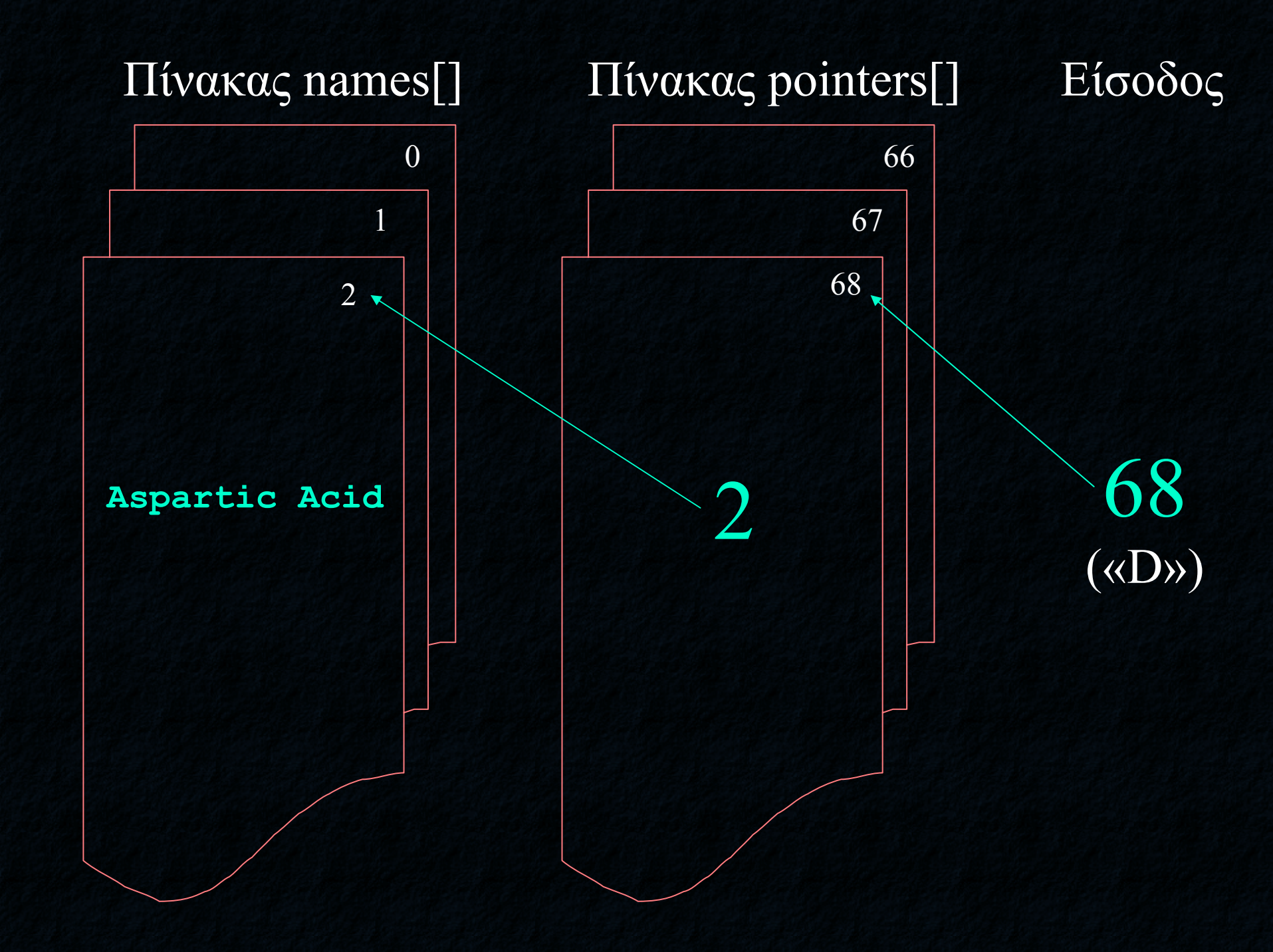
68

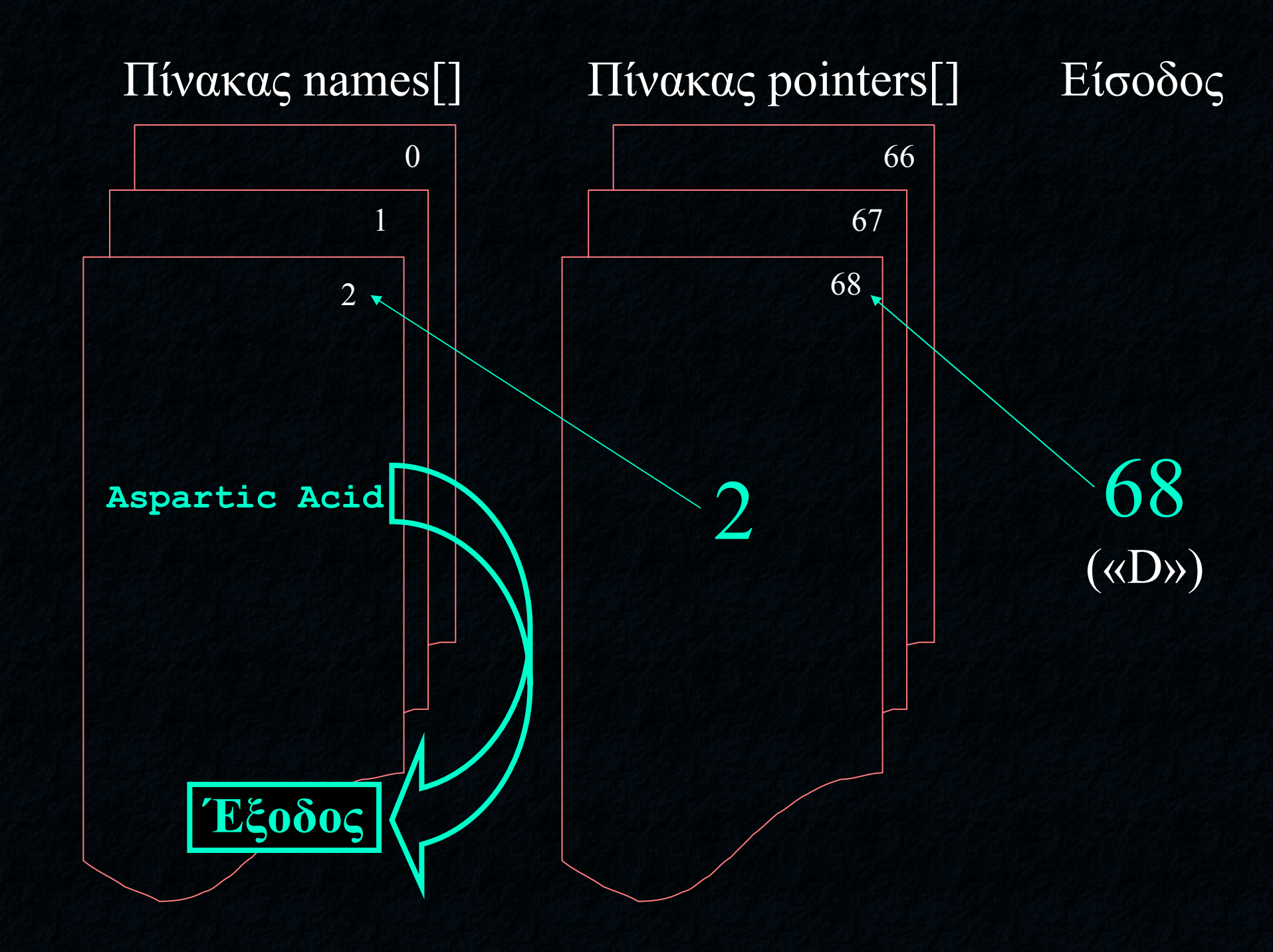
(«D»)


Πίνακας names[]
Πίνακας pointers[]
Είσοδος
0
1
2
66
67
68
68
(«D»)


Πίνακας names[]
Πίνακας pointers[]
Είσοδος
0
1
2
66
67
68
2
68
(«D»)


Πίνακας names[]
Πίνακας pointers[]
Είσοδος
0
1
2
66
67
68
2
68
(«D»)


Πίνακας names[]
Πίνακας pointers[]
Είσοδος
0
1
2
66
67
68
Aspartic Acid
2
68
(«D»)


Πίνακας names[]
Πίνακας pointers[]
Είσοδος
0
1
2
66
67
68
Aspartic Acid
2
68
(«D»)
Έξοδος