

Προγραμματισμός I (HY120)

Διάλεξη 11:
Δείκτες & Πίνακες



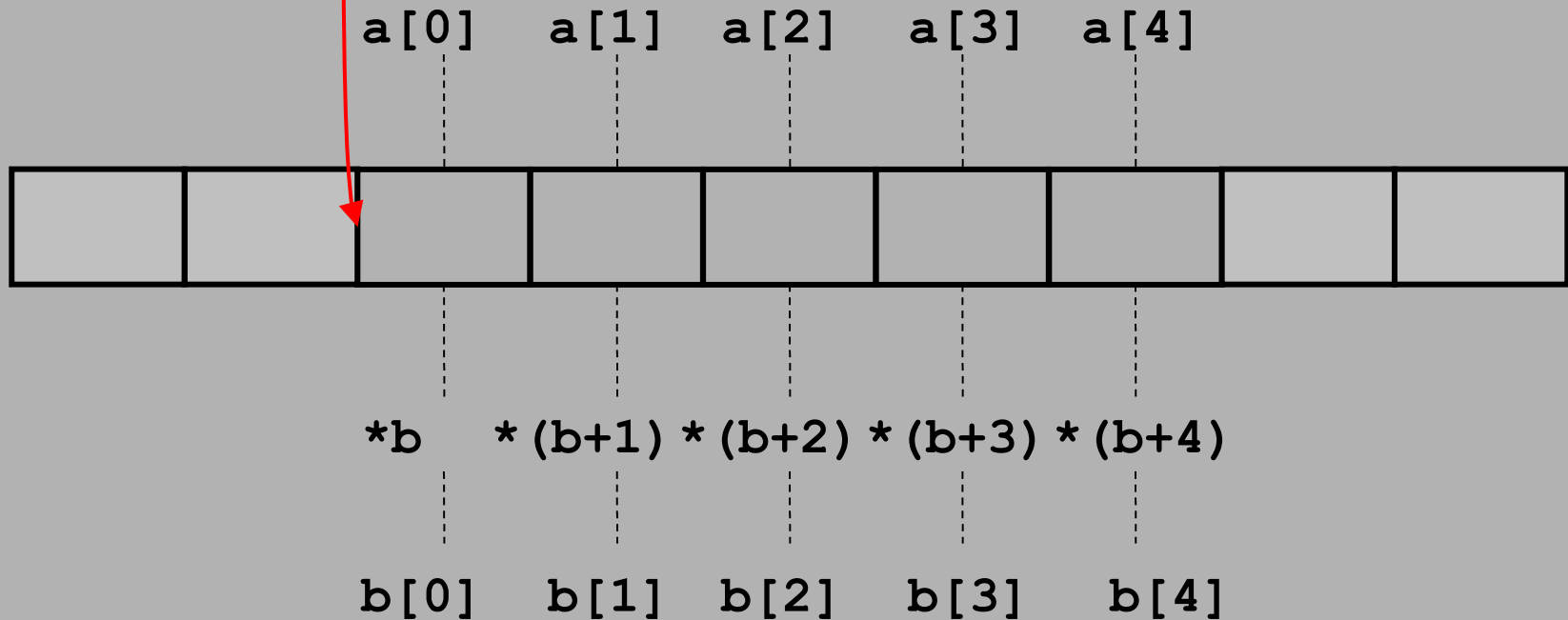
Δείκτες και πίνακες



- Μια μεταβλητή «μονοδιάστατος πίνακας από αντικείμενα 2 τύπου T» μπορεί να θεωρηθεί ως (είναι συντακτικά συμβατή με) μια μεταβλητή «δείκτης-σε-T» με **σταθερή τιμή** την **διεύθυνση** του **πρώτου στοιχείου** του πίνακα.
- Αντίστροφα, μια μεταβλητή δείκτης-σε-T (και μια διεύθυνση μεταβλητής τύπου T) μπορεί να θεωρηθεί ως η **αρχή** ενός μονοδιάστατου πίνακα από αντικείμενα τύπου T.
- Με χρήση δεικτών μπορεί να γίνει διέλευση των στοιχείων ενός πίνακα
 - Αντί της συμβατικής μεθόδου πρόσβασης μέσω της θέσης τους.
- Υπάρχει **συντακτική συμβατότητα** ανάμεσα σε μεταβλητές «πίνακας από T» και «δείκτης σε T».



```
int a[5];  
int * b = a;
```

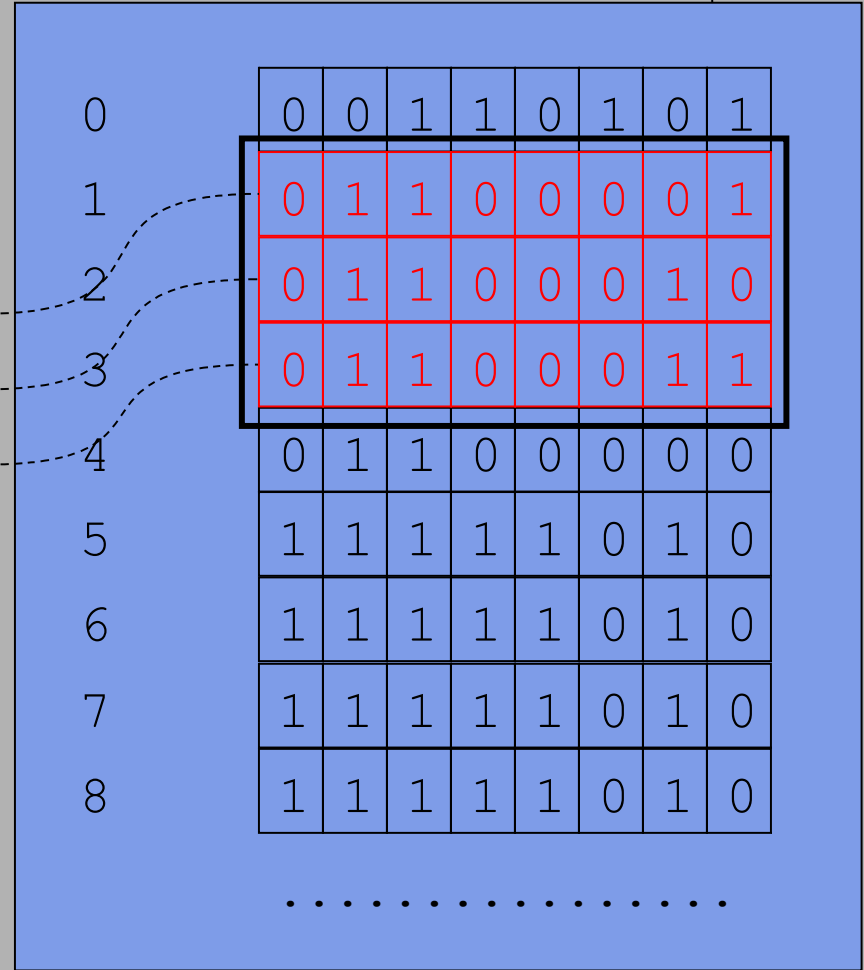




διεύθυνση περιεχόμενα

```
...  
char a[]={ 'a', 'b', 'c' };  
...
```

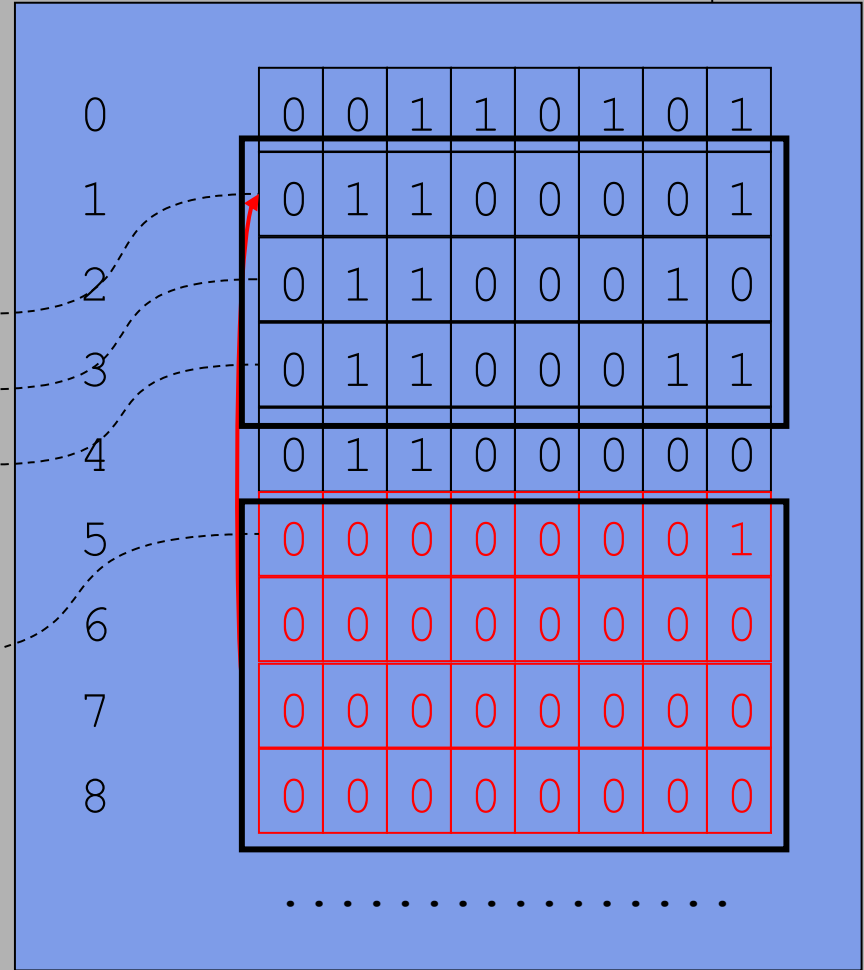
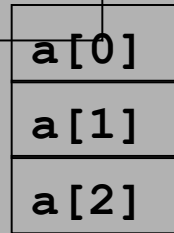
a[0]
a[1]
a[2]





διεύθυνση περιεχόμενα

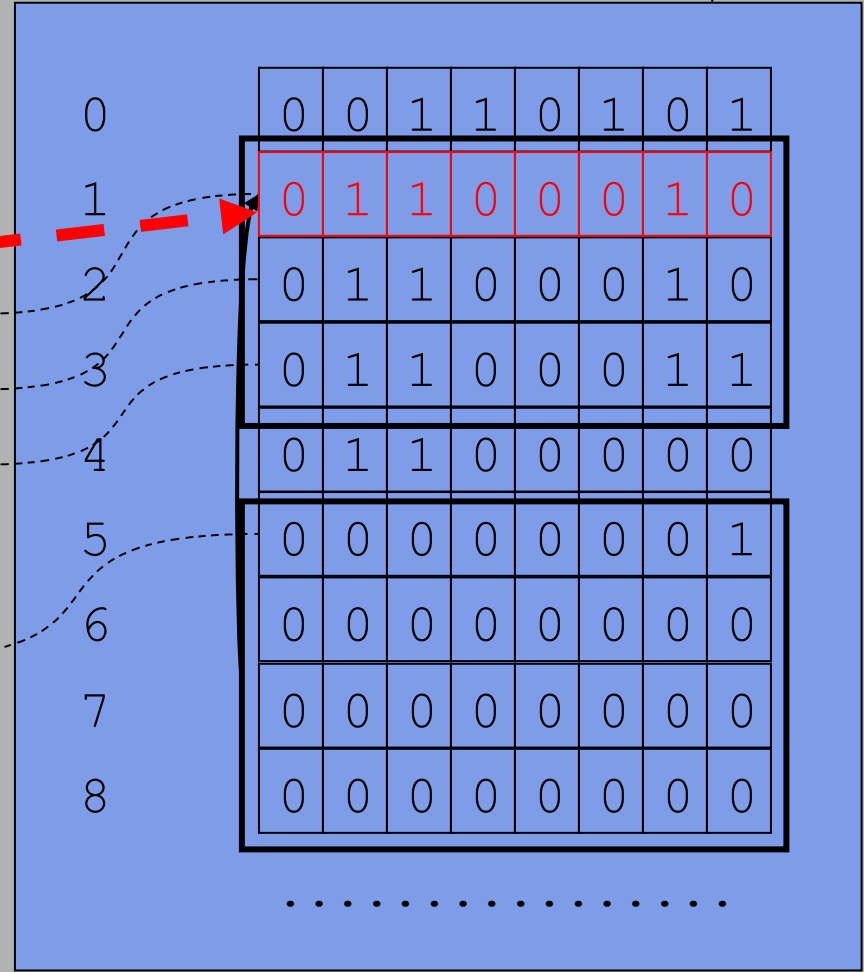
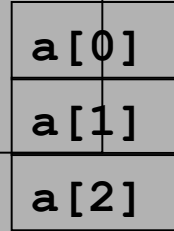
```
...  
char a[]={'a','b','c'};  
...  
char *b=a;
```





διεύθυνση περιεχόμενα

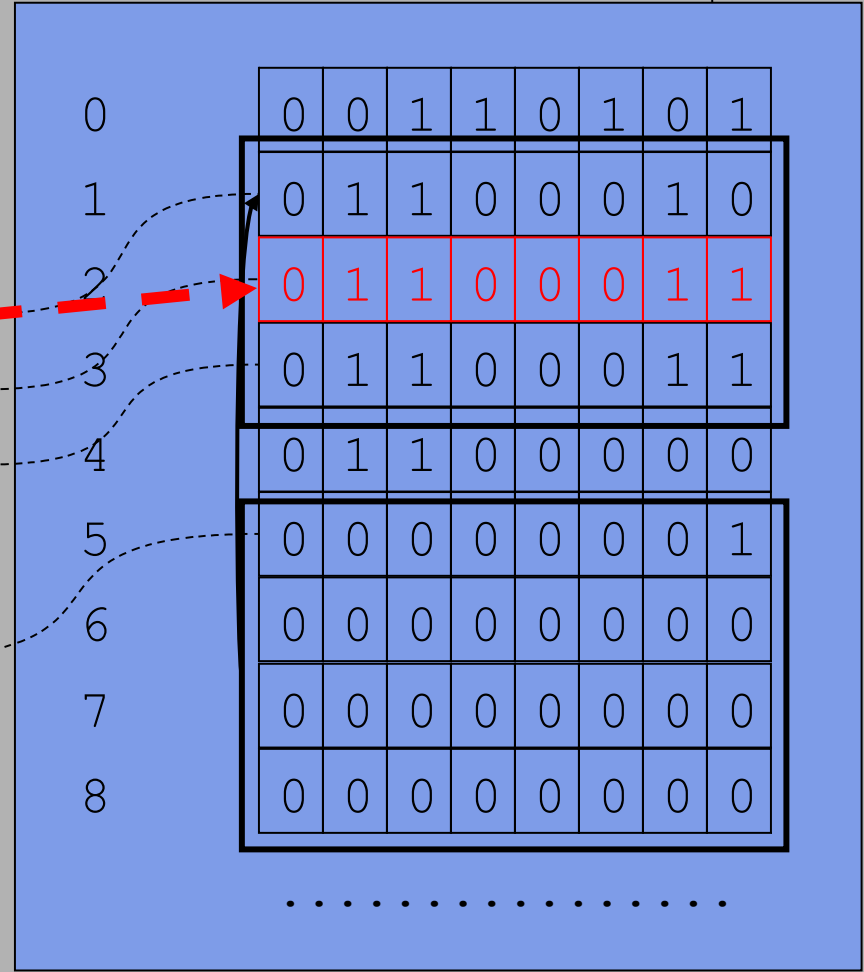
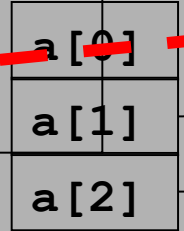
```
...  
char a[]={'a','b','c'};  
...  
char *b=a;  
  
b[0]++;
```





διεύθυνση περιεχόμενα

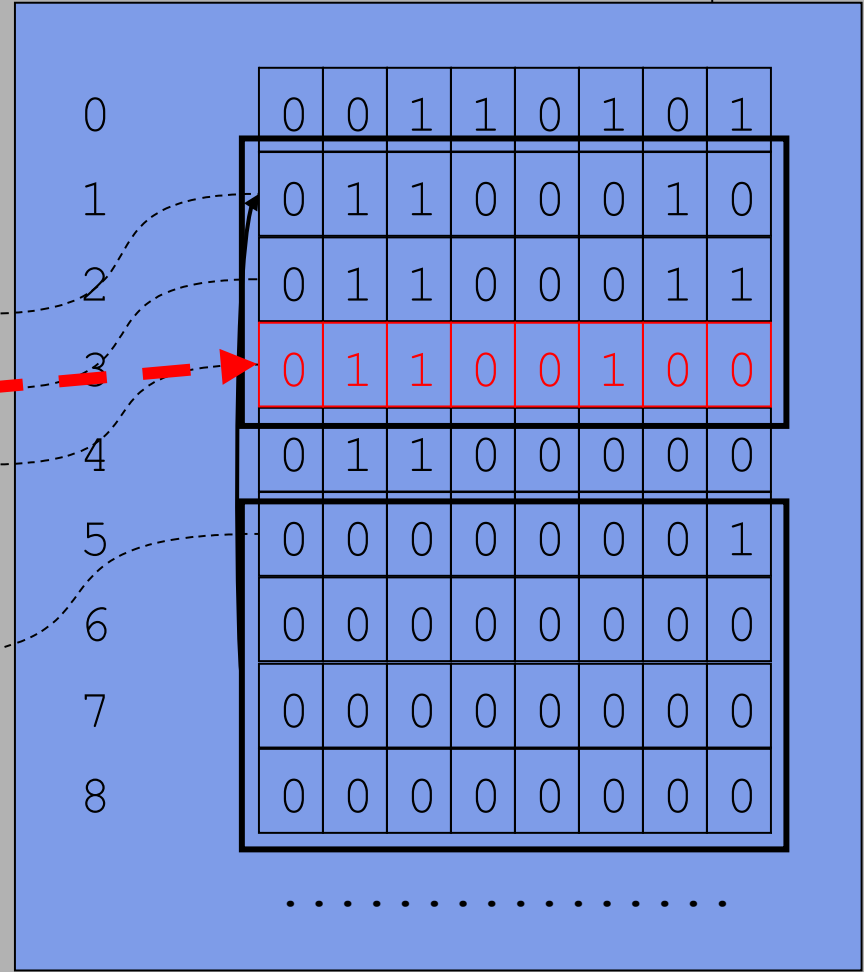
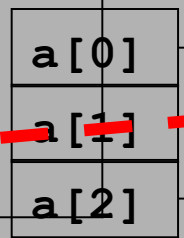
```
...  
char a[]={'a','b','c'};  
...  
char *b=a;  
  
b[0]++;  
b[1]++;
```





διεύθυνση περιεχόμενα

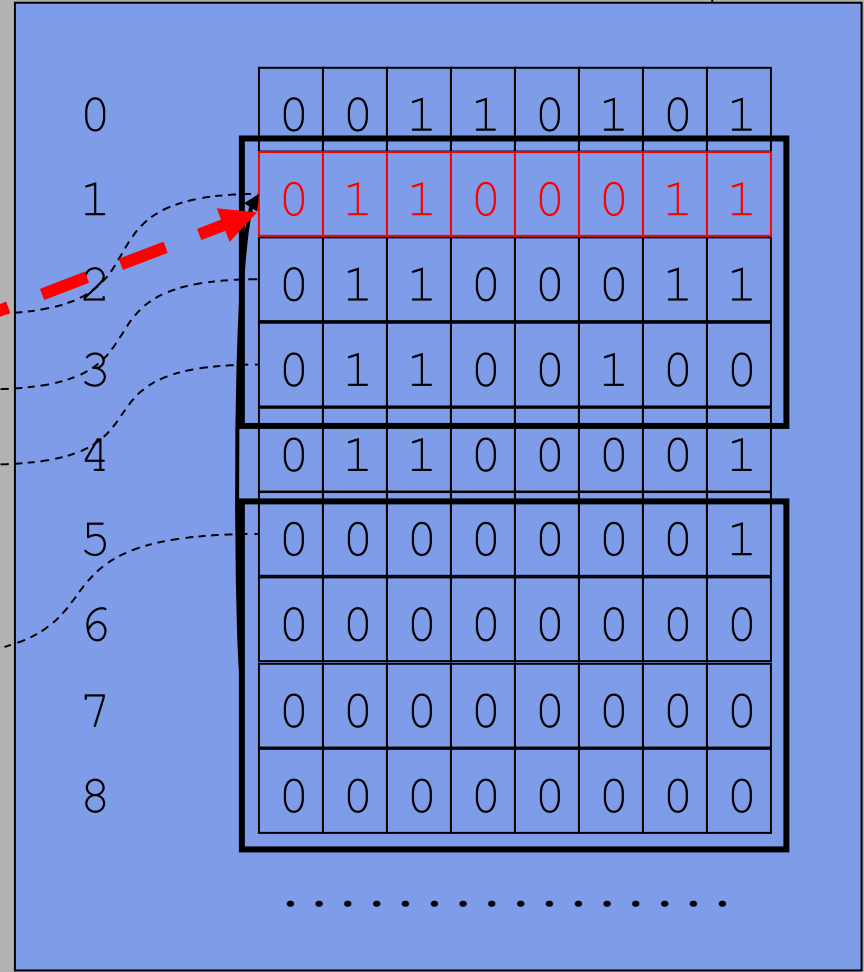
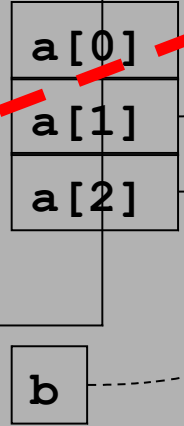
```
...  
char a[]={'a','b','c'};  
...  
char *b=a;  
  
b[0]++;  
b[1]++;  
b[2]++;
```





διεύθυνση περιεχόμενα

```
...  
char a[]={'a','b','c'};  
...  
char *b=a;  
  
b[0]++;  
b[1]++;  
b[2]++;  
  
*b=*b+1;
```





διεύθυνση περιεχόμενα

```
...  
char a[]={'a','b','c'};
```

```
...  
char *b=a;
```

```
b[0]++;
```

```
b[1]++;
```

```
b[2]++;
```

```
*b=*b+1;
```

```
b++;
```

a[0]
a[1]
a[2]

b

0

0	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---

1

0	1	1	0	0	0	1	1
---	---	---	---	---	---	---	---

2

0	1	1	0	0	0	1	1
---	---	---	---	---	---	---	---

3

0	1	1	0	0	1	0	0
---	---	---	---	---	---	---	---

4

0	1	1	0	0	0	0	1
---	---	---	---	---	---	---	---

5

0	1	1	0	0	0	0	1
---	---	---	---	---	---	---	---

6

0	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

7

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

8

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

.....



διεύθυνση περιεχόμενα

```
...  
char a[]={'a','b','c'};
```

```
...  
char *b=a;
```

```
b[0]++;
```

```
b[1]++;
```

```
b[2]++;
```

```
*b=*b+1;
```

```
b++;
```

```
*b=*b+1;
```

a[0]

a[1]

a[2]

b

0

0	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---

1

0	1	1	0	0	0	1	1
---	---	---	---	---	---	---	---

2

0	1	1	0	0	1	0	0
---	---	---	---	---	---	---	---

3

0	1	1	0	0	1	0	0
---	---	---	---	---	---	---	---

4

0	1	1	0	0	0	0	1
---	---	---	---	---	---	---	---

5

0	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

6

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

7

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

8

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

.....



διεύθυνση περιεχόμενα

```
...  
char a[]={'a','b','c'};
```

```
...  
char *b=a;
```

```
b[0]++;
```

```
b[1]++;
```

```
b[2]++;
```

```
*b=*b+1;
```

```
b++;
```

```
*b=*b+1;
```

```
b++;
```

a[0]
a[1]
a[2]

b

0

0	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---

1

0	1	1	0	0	0	1	1
---	---	---	---	---	---	---	---

2

0	1	1	0	0	1	0	0
---	---	---	---	---	---	---	---

3

0	1	1	0	0	1	0	0
---	---	---	---	---	---	---	---

4

0	1	1	0	0	0	0	1
---	---	---	---	---	---	---	---

5

0	0	0	0	0	0	1	1
---	---	---	---	---	---	---	---

6

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

7

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

8

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

.....

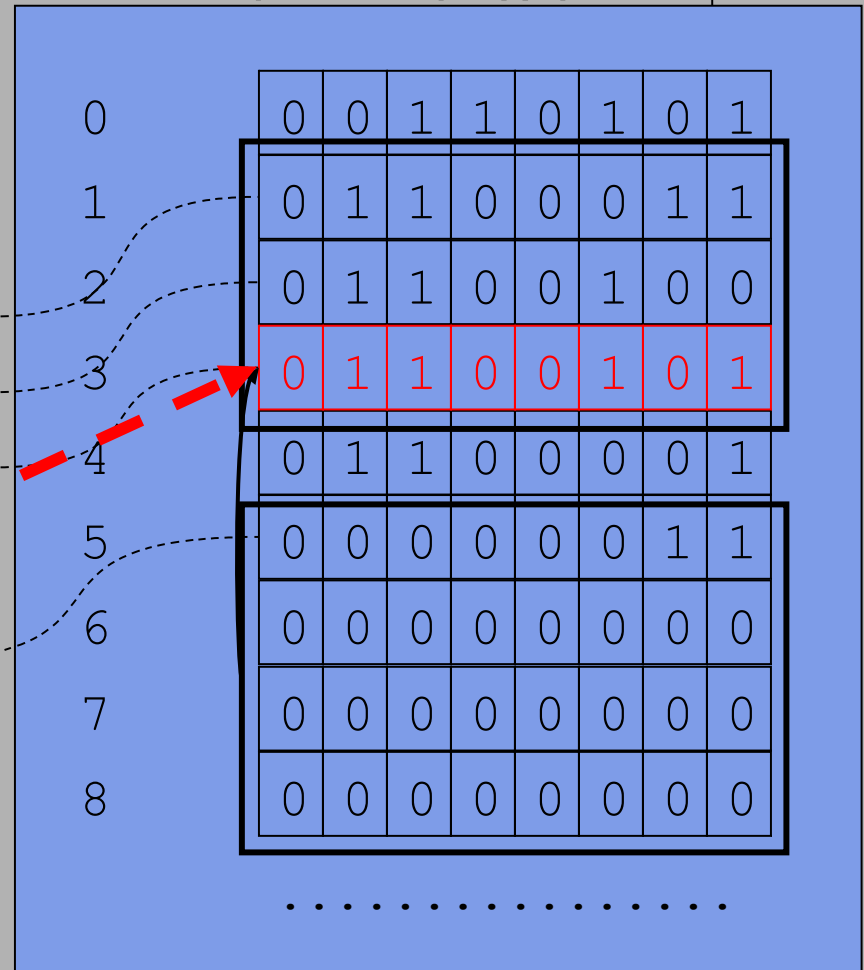


διεύθυνση περιεχόμενα

```
...  
char a[]={'a','b','c'};  
...  
char *b=a;  
  
b[0]++;  
b[1]++;  
b[2]++;  
  
*b=*b+1;  
b++;  
*b=*b+1;  
b++;  
*b=*b+1;
```

a[0]
a[1]
a[2]

b





```
/* εκτύπωση strings αποθηκευμένων σε ένα πίνακα */  
  
char str[] = {'o', 'n', 'e', '\0', 't', 'w', 'o', '\0', 'x'};  
char *str_p;  
  
printf("%s\n", str);          /* εκτυπώνει "one" */  
printf("%s\n", &str[0]);     /* εκτυπώνει "one" */  
  
str_p = str;  
printf("%s\n", str_p);      /* εκτυπώνει "one" */  
  
printf("%s\n", &str[4]);     /* εκτυπώνει "two" */  
  
str_p = str_p + 4;  
printf("%s\n", str_p);      /* εκτυπώνει "two" */  
printf("%s\n", str_p + 1);  /* εκτυπώνει "wo" */
```

Προσπέλαση Πίνακα με Δείκτες: Καλή ιδέα;



15

- Η πρόσβαση σε στοιχεία του πίνακα με δείκτη μπορεί να είναι πιο γρήγορη από την συμβατική πρόσβαση μέσω θέσης στον πίνακα
 - Γιατί;
 - Δεν είναι καλή ιδέα να χρησιμοποιείται, εκτός και αν υπάρχει σοβαρός λόγος, π.χ. η συγκεκριμένη πρόσβαση αποτελεί όντως σημείο συμφόρησης ενός ιδιαίτερα κρίσιμου κώδικα.
 - Σε γενικές γραμμές, η **συμβατική** πρόσβαση στα στοιχεία ενός πίνακα **βελτιώνει** την αναγνωσιμότητα του κώδικα.
- Η πρόσβαση με δείκτες είναι συνήθης τακτική ιδίως για προγραμματισμό σε «χαμηλό» επίπεδο συστήματος
 - π.χ. Λειτουργικό σύστημα.



```
/* διέλευση (εκτύπωση) πίνακα ακεραίων */  
  
int i,a[N];  
  
for (i=0; i<N; i++)  
    printf("%d ",a[i]);  
printf("\n");
```

```
/* διέλευση (εκτύπωση) πίνακα ακεραίων */  
  
int i,*a_ptr,a[N];  
  
a_ptr=a;  
for (i=0; i<N; i++)  
    printf("%d ",a_ptr[i]);  
printf("\n");
```

```
/* διέλευση (εκτύπωση) πίνακα ακεραίων */  
  
int *a_ptr,a[N];  
  
for (a_ptr=a; a_ptr<a+N; a_ptr++)  
    printf("%d ",*a_ptr);  
printf("\n");
```



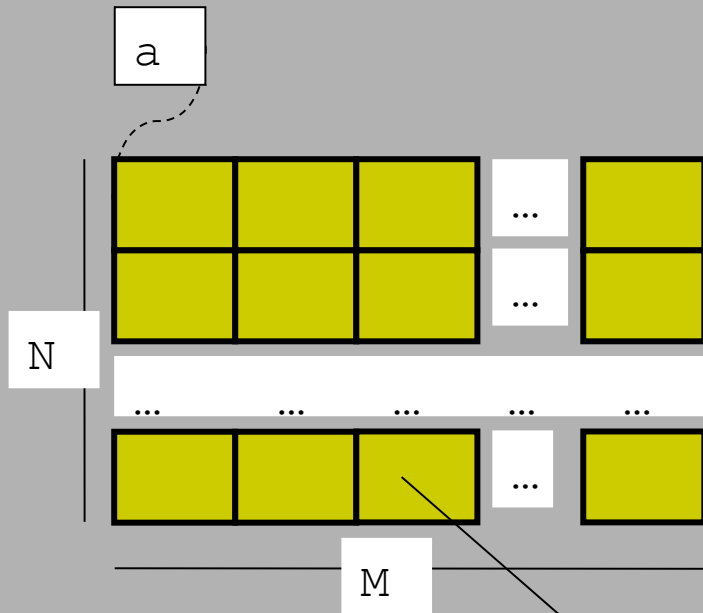

Πίνακες από δείκτες

- Οι πίνακες από δείκτες (π.χ. $*b[N]$) «μοιάζουν» με 2-διάστατους πίνακες (π.χ. $a[N][M]$), όμως:
 - Κάθε σειρά $a[i]$ του 2-διάστατου πίνακα a έχει ακριβώς τον ίδιο αριθμό στοιχείων (M).
 - Οι σειρές του πίνακα a αποθηκεύονται πάντα σε συνεχόμενες θέσεις μνήμης, η μία μετά την άλλη.
 - Κάθε δείκτης $b[i]$ του πίνακα b μπορεί να δείχνει σε εντελώς διαφορετική περιοχή μνήμης, η οποία να μην έχει καμία σχέση με τις περιοχές μνήμης όπου δείχνουν οι υπόλοιποι δείκτες του πίνακα b .
 - Τα δεδομένα που βρίσκονται αποθηκευμένα εκεί όπου δείχνουν τα στοιχεία του b , μπορεί να «ανήκουν» σε συμβατικές μεταβλητές του προγράμματος.



2-διάστατος πίνακας αντικειμένων τύπου T

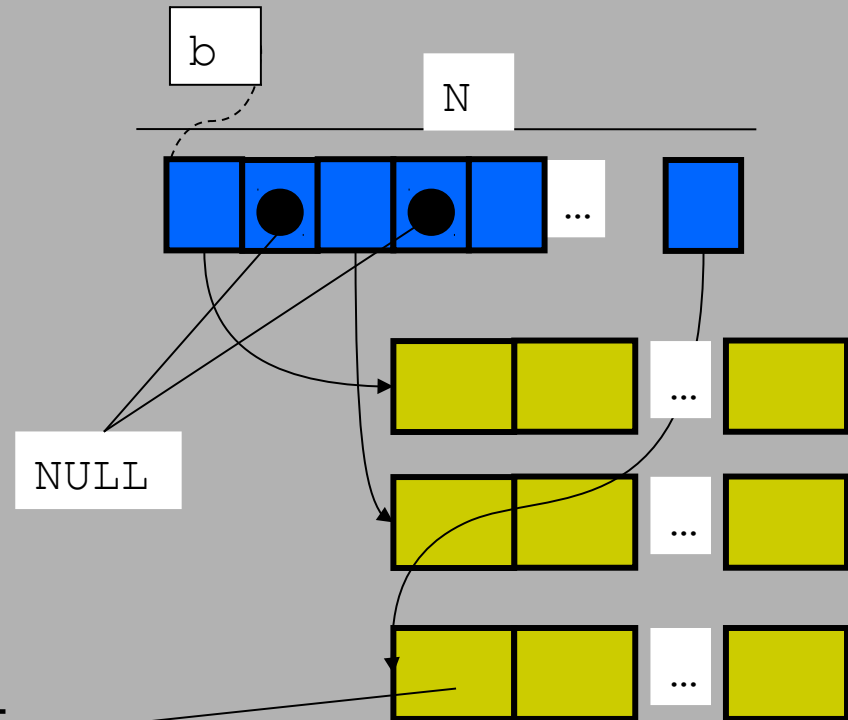
```
T a[N][M];
```



δεδομένα τύπου T

πίνακας από δείκτες σε αντικείμενα τύπου T

```
T *b[N];
```





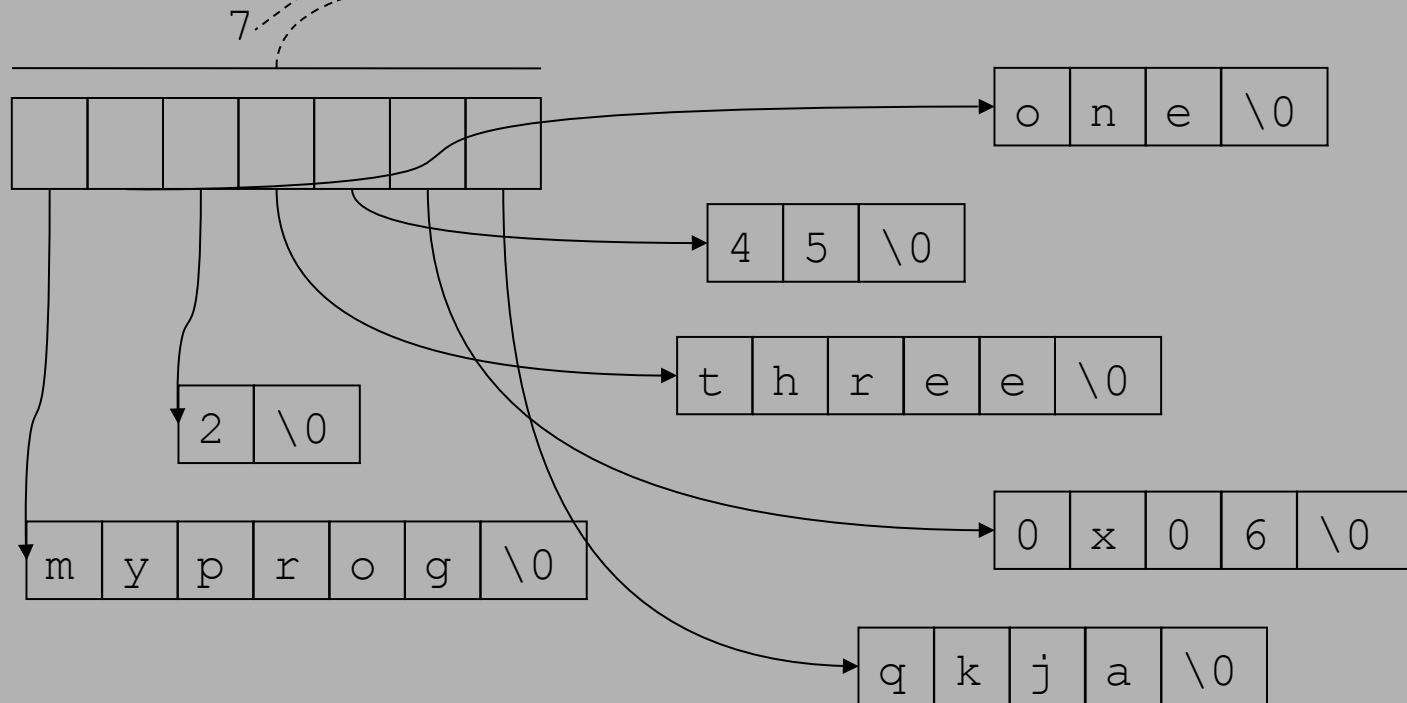
Παράμετροι της `main`

- Συχνά είναι βολικό το πρόγραμμα να δέχεται κάποια δεδομένα ως **παραμέτρους εκκίνησης** αντί να τα ζητά από τον χρήστη μέσω αντίστοιχου διαλόγου.
- Η συνάρτηση **`main`** δέχεται δύο παραμέτρους.
 1. Τον αριθμό των ορισμάτων **`argc`** (argument count) που περάστηκαν στο πρόγραμμα από το περιβάλλον εκτέλεσης, συμπεριλαμβανομένου του ονόματος του (όρισμα 0).
 2. Τον πίνακα από δείκτες-σε-χαρακτήρα **`argv`** (argument vector), όπου το *i*-οστό στοιχείο του πίνακα περιέχει ένα δείκτη σε θέση μνήμης όπου βρίσκεται αποθηκευμένο το *i*-οστό αλφαριθμητικό που δόθηκε ως όρισμα από την γραμμή εντολών.



```
>./myprog one 2 three 45 0x06 qkja<enter>
```

```
int main (int argc, char *argv[]) {  
...  
}
```



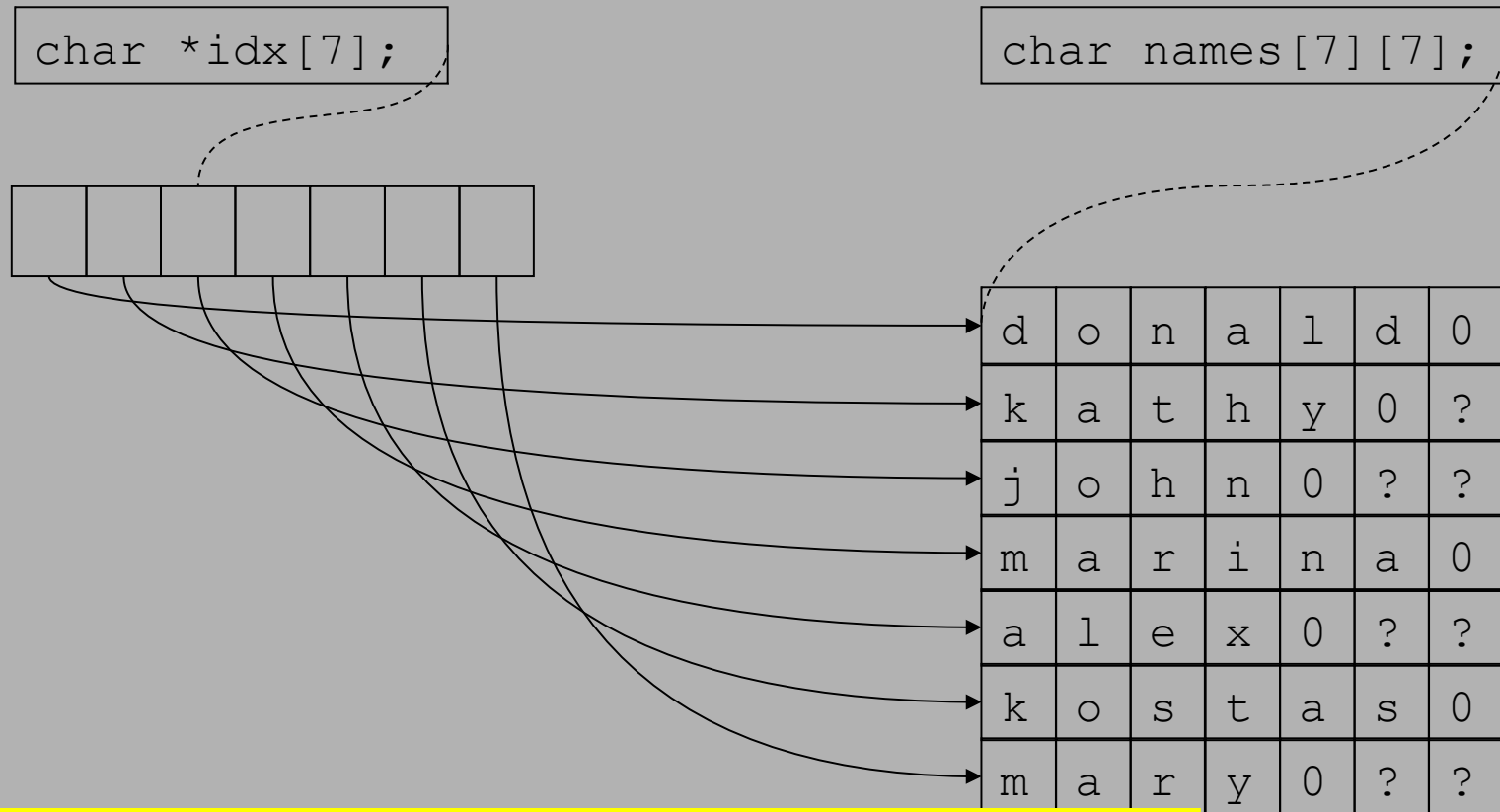
Ταξινόμηση με Ευρετήριο



21

πίνακας από N δείκτες
σε ονόματα στον πίνακα

Πίνακας από 7 ονόματα
(το πολύ 6 χαρακτήρων)



πριν την ταξινόμηση του ευρετηρίου

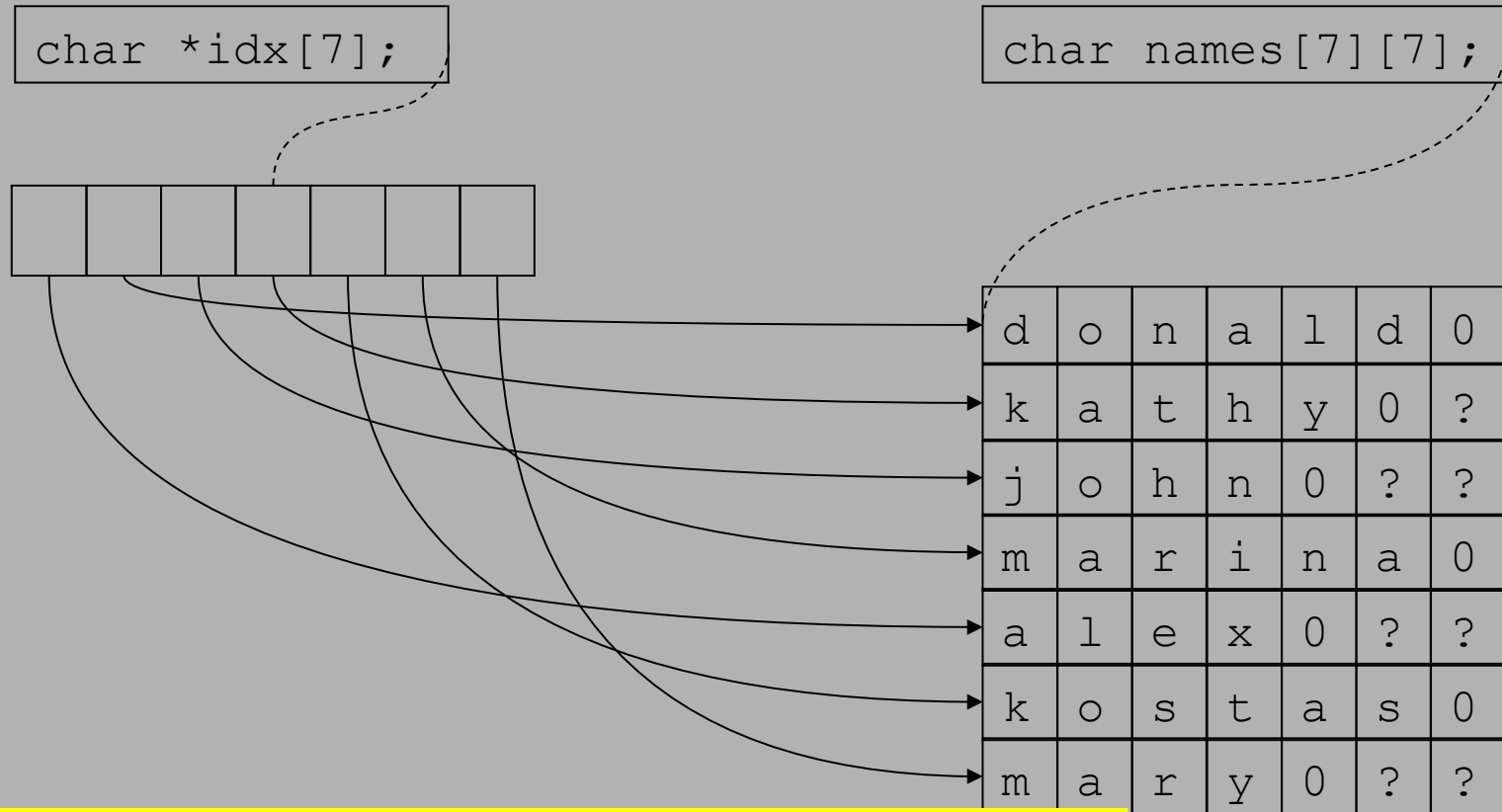
Ταξινόμηση με Ευρετήριο



22

πίνακας από N δείκτες
σε ονόματα στον πίνακα

Πίνακας από 7 ονόματα
(το πολύ 6 χαρακτήρων)



μετά την ταξινόμηση του ευρετηρίου

```
/* ταξινόμηση αλφαριθμητικών με ευρετήριο */
#include <stdio.h>
#include <string.h>
#define N 7
#define MAXNAMELEN 7
```



23

```
int main(int argc, char *argv[]) {
    char names[N][MAXNAMELEN], *idx[N], *tmp; int i, j;

    for (i=0; i<N; i++) {
        printf("enter name: "); scanf("%6s", names[i]);
    }

    for (i=0; i<N; i++) { idx[i] = names[i]; }

    for (i=0; i<N; i++) {
        for (j=i; j<N; j++) {
            if (strcmp(idx[i], idx[j])>0) {
                tmp = idx[i]; idx[i] = idx[j]; idx[j] = tmp;
            }
        }
    }

    for (i=0; i<N; i++) { printf("%s\n", idx[i]); }
    return (0);
}
```

συνάρτηση σύγκρισης
αλφαριθμητικών