ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

ΑΡΧΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Κεφάλαιο 10

Επιμέλεια:
Βασίλης Παλιουράς , Αναπληρωτής Καθηγητής
Ευάγγελος Δερματάς , Αναπληρωτής Καθηγητής
Σταύρος Νούσιας , Βοηθός Ερευνητή
Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών & Τεχνολογίας Υπολογιστών

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τυπου άδειας χρησης, η άδεια χρησης αναφερεται ρητώς.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Χρηματοδότηση

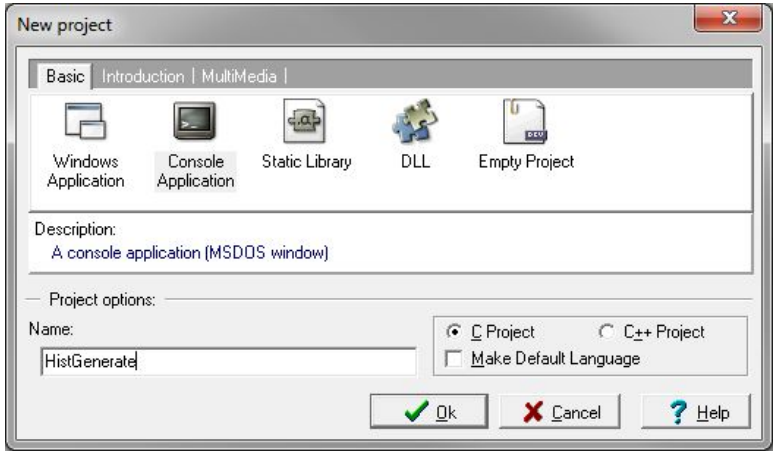
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου των διδασκόντων καθηγητών.
- Το έργο «Ανοικτά Ακαδήμαϊκά Μαθήματα στο Πανεπιστήμιο Πατρών» έχει χρηματοδοτηθεί μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκη Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικους πόρους.

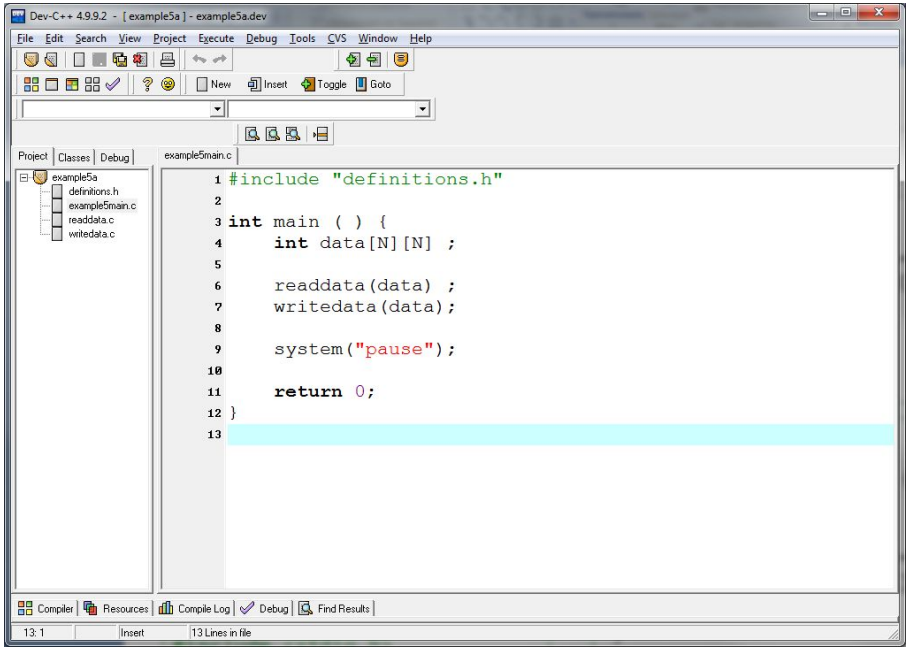
Ανάπτυξη

Το παρόν εκπαιδευτικό υλικό αναπτύχθηκε στο τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών του Πανεπιστημίου Πατρών

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Project σε Dev-C++





Δείκτες (Pointers)

- **Δείκτης**: μεταβλητή στην οποία αποθηκεύουμε **διεύθυνση** θέσης μνήμης.
  - δείχνει **που** είναι αποθηκευμένα δεδομένα
- **Δήλωση Δείκτη**

  `<τύπος>  *<όνομα δείκτη>;`
- ΠΡΟΣΟΧΗ: Το όνομα πίνακα **είναι** διεύθυνση, αλλά **δεν είναι** μεταβλητή!!!

Παράδειγμα δήλωσης δείκτη

- `char *ch_ptr;`
- Η μεταβλητή ch_ptr περιέχει ***διεύθυνση μνήμης*** στην οποία είναι αποθηκευμένο δεδομένο τύπου χαρακτήρα.

- `char ch;`
  - Η μεταβλητή ch έχει ως αξία χαρακτήρα.

- Μπορούμε να δηλώσουμε δείκτες σε δεδομένα διαφόρων τύπων
  - Βασικών τύπων
  - Κατασκευασμένων τύπων

Παράδειγμα χρήσης δείκτη

```
void main ( ) {
char ch = 'a', ch2;
char *ch_ptr ;

ch_ptr = &ch ;
ch2 = *ch_ptr ;

printf ("%c", ch2);

}
```

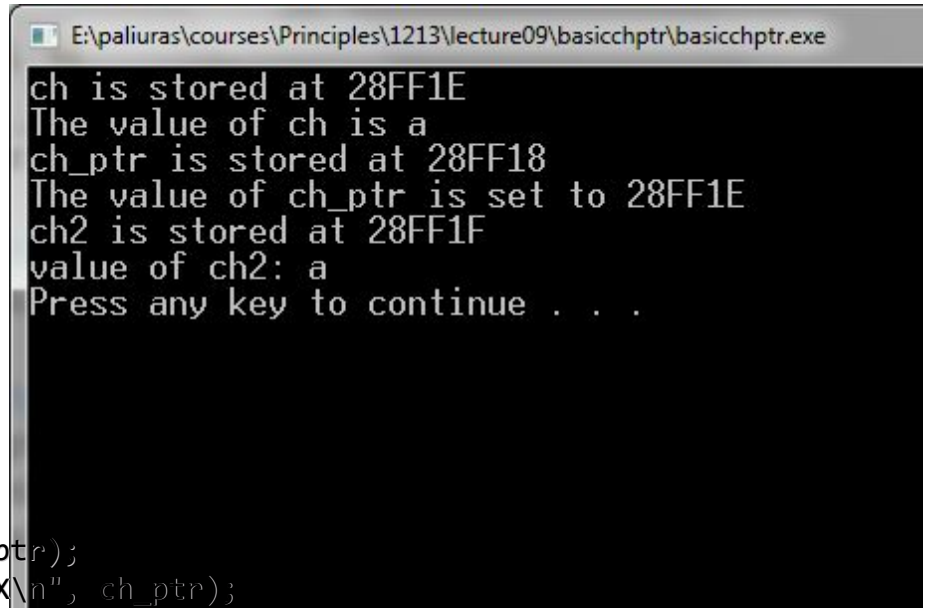
Δήλωση δείκτη σε χαρακτήρα

* : **Περιεχόμενα της θέσης** στην οποία δείχνει ο δείκτης

Πίνακες και δείκτες

- `int arr[10], n ;`
- `*(arr + n)` ⇔ `arr[n]`
- `arr + n` ⇔ `&arr[n]`
- …αλλά και `n[arr]` ⇔ `*(n+arr)` // (!!!)
- χρησιμοποιούμε δείκτες για να περάσουμε ως όρισμα σε συνάρτηση πίνακες
  - ακριβέστερα: σε ποια διεύθυνση μνήμης βρίσκεται το πρώτο στοιχ είο του πίνακα.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    char ch, ch2;
    char *ch_ptr ;

    ch = 'a';
    printf( "ch is stored at %X\n", &ch);
    printf( "The value of ch is %c\n", ch);

    //ch_ptr is set to point to ch
    ch_ptr = &ch ;
    printf( "ch_ptr is stored at %X\n", &ch_ptr);
    printf( "The value of ch_ptr is set to %X\n", ch_ptr);

    //contents of ch_ptr (i.e., value of ch)
    //are copied to ch2
    ch2 = *ch_ptr ;
    printf ("ch2 is stored at %X\n", &ch2);
    printf ("value of ch2: %c\n", ch2);

    system("PAUSE");
    return 0;
```



Χάρτης μνήμης

Διεύθυνση	Περιεχόμενα μνήμης σε hex	Όνομα μεταβλητής
28FF18	1E	ch_ptr
28FF19	FF	
28FF1A	28	
28FF1B	00	
28FF1C		
28FF1E	61	ch
28FF1F	61	ch2

Η συνάρτηση `getchar()`

- Διαβάζει έναν χαρακτήρα από την είσοδο.

12

Μερικές συναρτήσεις βασικής βιβλιοθήκης για αλφαριθμητικά

- πρότυπα στο <string.h>
- `char *strcpy (char *, const char *) ;`
- `int strcmp (const char *, const char *) ;`
- `char *strcat (char *, const char *) ;`
- `char *strchr (const char *, char) ;`
- `size_t strlen (const char *) ;`
- …

Συναρτήσεις για προκαθορισμένο πλήθος χαρακτήρων - παραδείγματα

```
#include <string.h>

int strncmp(const char *s1, const char
*s2, size_t n);


char *strncpy(char *s1, const char *s2,
size_t n);
```

Παράδειγμα

- Διάβασε ένα αλφαριθμητικό
- Μέτρησε πόσες φορές περιλαμβάνει τον χ αρακτήρα ‘a’
- Τύπωσε το αποτέλεσμα.

```
void readstring(char *) ;
int countA(char *);
void printresult( int ) ;
```

Η main ( ) του παραδείγματος

```
main ( ) {
 char astring[N];
 int acount ;

 readstring (astring) ;

 acount =
    countA(astring);

 printresult(acount) ;
}
```

```
#define N 50
#include <stdio.h>
void readstring(char *) ;
int countA(char *);
void printresult( int ) ;
```

```
void readstring(char *s ) {
        printf ("alpharithmitiko? ");
        scanf("%s", s);
}
```

```
void printresult ( int a) {
        printf("The result is %d\n", a);
}
```

Υλοποίηση της **int** countA(**char** *);

```
int countA(char *s) {
int count = 0 ;
int i = 0;
  while ( s[i] != 0 ) {
    if (s[i] == 'a')
       count ++;
    i ++ ;
    }
  return count ;
}
```

γιατί το μηδέν δηλώνει τέλος του αλφαριθμητικού

αν ο τρέχων χαρακτήρας είναι ίσος με ‘a’, αύξησε το μετρητή count κατά ένα

προχώρησε στον επόμενο χαρακτήρα του αλφαριθμητικού

Πρόθεμα και Επίθεμα prefix και postfix)

- i++; /*postfix */
- ++ i; /* prefix */
- i = 0;
- myprint(i++);
- i = 0;
- myprint(++i);

Η myprint ( ) καλείται με διαφορετικό όρισμα (διαφορετική τιμή) στις δύο περιπτώσεις!

Πρόβλημα

- Πώς θα γράφαμε πρότυπο και τον ορισμό συνάρτησης η οποία θα δέχεται ως ορίσματα:

  α) το αλφαριθμητικό και

  β) τον χαρακτήρα για τον οποίο γίνεται ο έλεγχος.

- Πρότυπο αυτής:

  int countchar(**char** *, **char**);

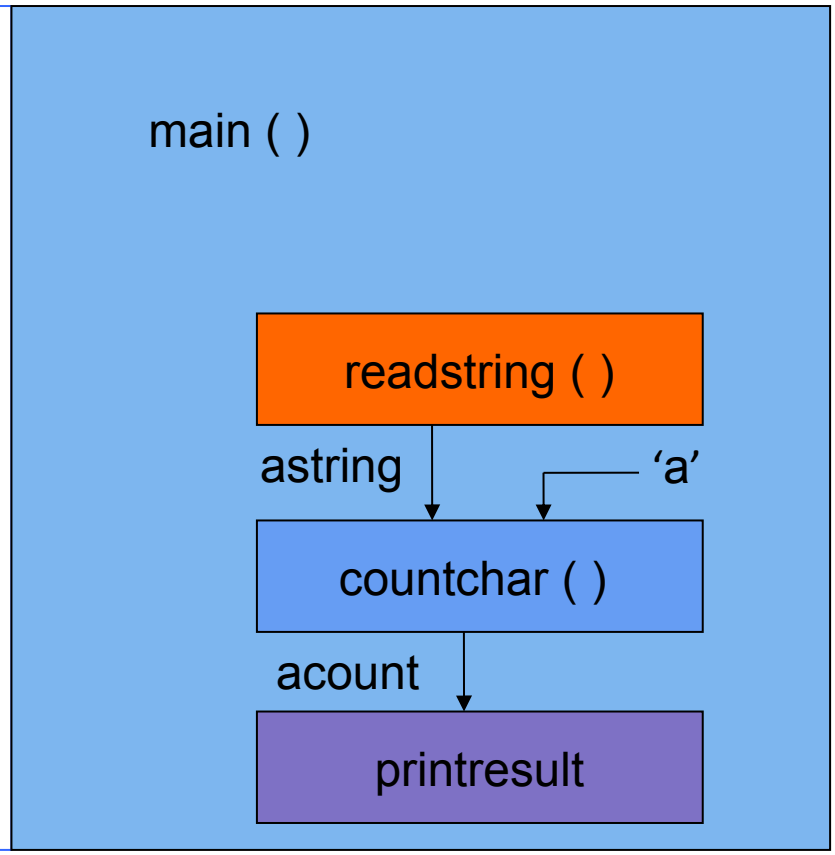
- Παράδειγμα κλήσης

  acount = countchar(astring, 'a');

- Τι πλεονεκτήματα έχει να γράφουμε γενικότερο κώδικα;

Χρήση της countchar(**char** *, **char**)

```
main ( ) {
 char astring[N];
 int acount ;

 readstring (astring) ;

 acount = countchar(astring, ‘a’);

 printresult(acount) ;
}
```



Πιθανές υλοποιήσεις

```
int countA(char *s, char ch) {
int count = 0 ;
int i = 0;
  while ( s[i] != 0 ) {
     if (s[i] == ch)
        count ++;
     i ++ ;
     }
  return count ;
}
```

```
int countA(char *s, char ch) {
int count = 0 ;
int i ;
  for (i=0; s[i] != 0; i++)
     if (s[i] == ch)
                count ++;
   return count ;
}
```

int countchar(**char** *, **char**);

```
int countchar(char *s, char c) {
int count = 0 ;
int i = 0;
  while ( s[i] ) {
                count += (s[i] == c);
              i ++ ;
              }
  return count ;
}
```

```
int countchar(char *s, char c) {
int count = 0 ;
int i = 0;
  while ( s[i] )
                count += (s[i++] == c);
return count ;
}
```

postfix notation

int countchar(**char** *, **char**);

```
int countchar(char *s, char c) {
int count = 0 ;
int i = 0;
  while ( s[i] ) {
                count += (s[i] == c);
              i ++ ;
              }
  return count ;
}
```

```
int countchar(char *s, char c) {
int count = 0 ;
int i = 0;
  while ( s[i] )
                count += (s[i++] == c);
return count ;
}
```

postfix notation

```
int countchar(char *s, char c) {
int count = 0 ;
int i ;

  for( i = 0; s[i]; count += (s[i++] == c));

return count ;
}
```

```
int countchar(char *s, char c) {
int count = 0 ;
int i = 0;

  for( ; s[i]; count += (s[i++] == c));

return count ;
}
```

Πίνακας αλφαριθμητικών

```
#define N 3
#include <stdio.h>

main ( ) {
 int i ;
 char text[N][11] ={"dokimi", "test", "paradeigma"};

 printf("text requires %d bytes\n", sizeof text ) ;

 for (i=0 ; i < N ; printf ("%s ", text[i++]));

}
```

Ο πίνακας 2-D λειτουργεί ως πίνακας 1-D με στοιχεία πίνακες 1-D.

text[0]	'd'	'o'	'k'	'i'	'm'	'i'	0				
text[1]	't'	'e'	's'	't'	0						
text[2]	'p'	'a'	'r'	'a'	'd'	'e'	'i'	'g'	'm'	'a'	0

Παράδειγμα

```
#include <stdio.h>
#define CHARSPERWORD 10

void displaywords (char [][CHARSPERWORD], int);
void displaywordsptr (char *, int, int);
void displaywordsstr(char *, int , int ) ;

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

            char words[10][CHARSPERWORD] ={"hello",
"there"};
            char morewords[20][CHARSPERWORD] = {"a",
"few", "more", "words"};

            displaywords(words, 2);
            printf("\n");
            displaywords(morewords, 4);
            printf("\n");
            displaywordsptr((char *) morewords, 4,
CHARSPERWORD);
            printf("\n");
            displaywordsstr((char *) morewords, 4,
CHARSPERWORD);

            return 0;
}
```

```
void displaywords(char a[][CHARSPERWORD], int rows) {
            int i, j;
            for (i=0; i< rows ; i++)
                        for (j=0; j< CHARSPERWORD; j++)
                                    printf("%c",
a[i][j]);
}

void displaywordsptr(char *a, int rows, int columns) {
            int i, j;
            for (i=0; i<rows; i++)
                         for (j =0 ; j<columns; j++ )
                                    printf("%c",
*(a+columns*i+j));
}

void displaywordsstr(char *a, int rows, int columns) {
             int i;
              for (i=0; i<rows; i++)
                        printf("%s ", a + columns * i);
}
```

Παράδειγμα

```
#define N 3
#include <stdio.h>

main ( ) {
 int i ;
 char text1[N][11] ={"dokimi", "test", "paradeigma"};
 char *text2[N] = {"dokimi", "test", "paradeigma"};

 printf("text1 requires %d bytes\n", sizeof text1 ) ;
 printf("text2 requires %d bytes\n", sizeof text2 );

for (i=0 ; i < N ; printf ("%s ", text2[i++]))
    ;

}
```

Πίνακας 1-D με στοιχεία δείκτες σε χαρακτήρα

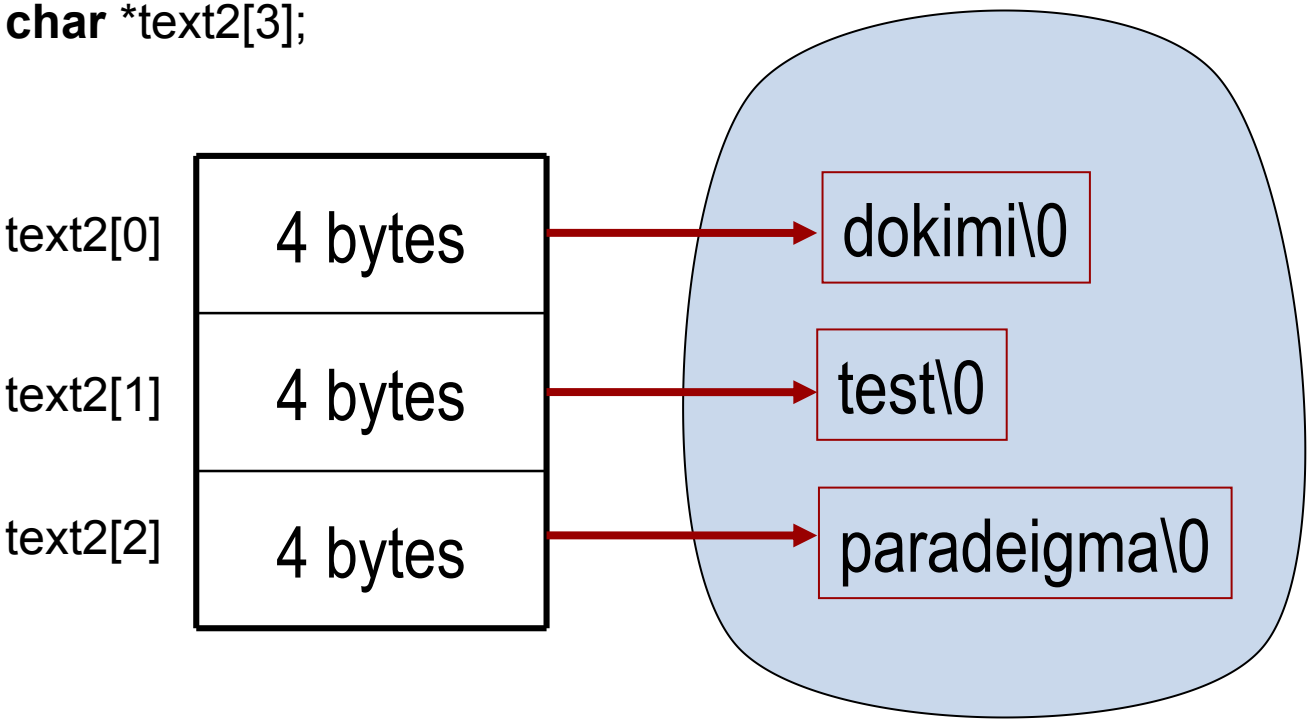
Προσοχή:
Δεν περιλαμβάνει τις αλφαριθμητικές σταθερές!

```
$ ./a
text1 requires 33 bytes
text2 requires 12 bytes
dokimi test paradeigma
Paliuras@MOB ~/nlect9
$
```

Οργάνωση μνήμης και πίνακες δεικτών σε χαρακτήρες

Οι αλφαριθμητικές σταθερές της αρχικοποίησης αποθηκεύονται σε άλλη περιοχή μνήμης.

char *text2[3];



text2: περιλαμβάνει τρεις διευθύνσεις, όχι τα ίδια τα αλφαριθμητικά.

Δήλωση και αρχικοποίηση

```
main ( ) {
    char name[10] = "katerina";

    printf ("%s", name);

    scanf("%s", name);

    printf ("%s", name);

}
```

Δήλωση και ανάθεση τιμής σε αλφαριθμητικό

```
char *strcpy(const char *, const char*);
```

```
main ( ) {
   char name[10];

   name = "katerina";

   printf ("%s", name);

   scanf("%s", name);

   printf ("%s", name);

}
```

Λάθος (στη C)

```
#include <string.h>
main ( ) {
   char name[10];

   strcpy(name,"katerina");
   printf ("%s", name);

   scanf("%s", name);
   printf ("%s", name);

}
```

Σωστός τρόπος:

Τι γίνεται με δείκτες;

```c
#include <stdio.h>
#include <string.h>

main ( ) {
char *name = "katerina";

printf ("%s", name);


}
```

```c
#include <stdio.h>
#include <string.h>

main ( ) {
char *name ;

name = "katerina";

printf ("%s", name);

}
```

```c
#include <stdio.h>
#include <string.h>

main ( ) {
char *name ;

name = "katerina";

printf ("%s", name);

*name = 'K';

printf ("%s", name);

}
```

σφάλμα χρόνου εκτέλεσης (run-time error):
segmentation fault

Λύση: χρήση διαθέσιμης περιοχής μνήμης (πχ με calloc ())

Σημείωμα αναφοράς

- Copyright Πανεπιστήμιο Πατρών, Παλιουράς Βασίλειος , Δερματάς Ευάγγελος «Αρχές Προγραμματισμού ». Έκδοση: 1.0. Πάτρα 2015
- Διαθέσιμο από τη δικτυακιακή διέυθυνση https://eclass.upatras.gr/modules/