

**MÉS BONUS**

# Què passa si declaro un string massa petit?



```
#include <stdio.h>
#define LLARG 5

int main(){
    char s[LLARG] = "abcdefghijkl"; /* Òbviament aquest string no hi cabrà */
    printf("String es: %s\n", s);
    return 0;
}
```

# Què passa si declaro un string massa petit?



```
#include <stdio.h>
#define LLARG 5

int main(){
    char s[LLARG] = "abcdefghijk"; /* Òbviament aquest string no hi cabrà */
    printf("String es: %s\n", s);
    return 0;
}
```

```
stringprova.c: In function 'main':
stringprova.c:5:21: warning: initializer-string for array of chars is too long
    char s[LLARG] = "abcdefghijklmnopq";
```

# Què passa si declaro un string massa petit?



```
#include <stdio.h>
#define LLARG 5

int main(){
    char s[LLARG] = "abcdefghijk"; /* Òbviament aquest string no hi cabrà */
    printf("String es: %s\n", s);
    return 0;
}
```

```
stringprova.c: In function 'main':
stringprova.c:5:21: warning: initializer-string for array of chars is too long
    char s[LLARG] = "abcdefghijklmnopq";
```

**Si passo de tot i l'executo...?**

# Què passa si declaro un string massa petit?



```
#include <stdio.h>
#define LLARG 5

int main(){
    char s[LLARG] = "abcdefghijk"; /* Òbviament aquest string no hi cabrà */
    printf("String es: %s\n", s);
    return 0;
}
```

```
stringprova.c: In function 'main':
stringprova.c:5:21: warning: initializer-string for array of chars is too long
    char s[LLARG] = "abcdefghijklmnopq";
```

**Si passo de tot i l'executo...?**

```
String es: abcde
```

# Què passa si declaro un string massa petit?

```
● ● ●  
  
#include <stdio.h>  
#define LLARG 5  
  
int main(){  
    char s[LLARG] = "abcdefghijk"; /* Òbviament aquest string no hi cabrà */  
    printf("String es: %s\n", s);  
    return 0;  
}
```

```
stringprova.c: In function 'main':  
stringprova.c:5:21: warning: initializer-string for array of chars is too long  
    char s[LLARG] = "abcdefghijklmnopq";
```

**Si passo de tot i l'executo...?**

```
String es: abcde
```


**Creieu que m'ha posat el final de string?**

# Què passa si declaro un string massa petit?

Creieu que m'ha posat el final de string?

# Què passa si declaro un string massa petit?

Creieu que m'ha posat el final de string?



```
#include <stdio.h>
#define LLARG 5

int main(){
    int i;
    /* Òbviament aquest string no hi cap */
    char s[LLARG] = "abcdefghijklmnopq";
    i = 0;
    while(s[i] != '\0'){
        printf("Posicio i=%d "
               "el character es %c\n",
               i, s[i]);
        i++;
    }
    return 0;
}
```



# Què passa si declaro un string massa petit?

Creieu que m'ha posat el final de string?



```
#include <stdio.h>
#define LLARG 5


int main(){
    int i;
    /* Òbviament aquest string no hi cap */
    char s[LLARG] = "abcdefghijklmnopq";
    i = 0;
    while(s[i] != '\0'){
        printf("Posicio i=%d "
               "el caracter es %c\n",
               i, s[i]);
        i++;
    }
    return 0;
}
```

```
Posicio i=0 el caracter es a
Posicio i=1 el caracter es b
Posicio i=2 el caracter es c
Posicio i=3 el caracter es d
Posicio i=4 el caracter es e
Posicio i=5 el caracter es
```

# Què passa si declaro un string massa petit?

Creieu que m'ha posat el final de string?

```


#include <stdio.h>
#define LLARG 5

int main(){
    int i;
    /* Òbviament aquest string no hi cap */
    char s[LLARG] = "abcdefghijklmnopq";
    i = 0;
    while(s[i] != '\0'){
        printf("Posicio i=%d "
               "el caracter es %c\n",
               i, s[i]);
        i++;
    }
    return 0;
}
```

```

Posicio i=0 el caracter es a
Posicio i=1 el caracter es b
Posicio i=2 el caracter es c
Posicio i=3 el caracter es d
Posicio i=4 el caracter es e
Posicio i=5 el caracter es
```

**SÍ !!! Això vol dir que... la variable que  
estava en memòria després de la taula...  
HA QUEDAT SOBREESCRITA.**

# Què passa si declaro un string massa petit?

Creieu que m'ha posat el final de string?

```

#include <stdio.h>
#define LLARG 5

int main(){
    int i;
    /* Òbviament aquest string no hi cap */
    char s[LLARG] = "abcdefghijklmnopq";
    i = 0;
    while(s[i] != '\0'){
        printf("Posicio i=%d "
               "el caracter es %c\n",
               i, s[i]);
        i++;
    }
    return 0;
}
```

```


Posicio i=0 el caracter es a
Posicio i=1 el caracter es b
Posicio i=2 el caracter es c
Posicio i=3 el caracter es d
Posicio i=4 el caracter es e
Posicio i=5 el caracter es
```

**SÍ !!! Això vol dir que... la variable que estava en memòria després de la taula... HA QUEDAT SOBREESCRITA.**

**MORALEJA: Mai executar codis amb warnings... realment ens està avisant que alguna cosa va malament.**

# Què passa si declaro un string massa petit?

I si fem servir la funció scanf?



```
#include <stdio.h>
#define LLARG 5

int main(){
    int i;
    char s[LLARG];

    printf("Usuari, introdueix un string de llargada màxima %d\n",LLARG-1);
    printf("Sisplau sisplau fes-me cas que sinó no sé què passarà!\n");
    scanf("%s",s);
    printf("Has introduit: %s\n",s);
    return 0;
}
```

# Què passa si declaro un string massa petit?

**I si fem servir la funció scanf?**

```
MacPepino:Desktop clara$ ./stringprova
Usuari, introduceix un string de llargada màxima 4
Sisplau sisplau fes-me cas que sinó no sé què passarà!
soc_dolent
Has introduït: soc_dolent
```

**Ai Déu meu, m'ha sobreescrit les  
variables següents... sort que no devien  
ser importants**

# Què passa si declaro un string massa petit?

**I si fem servir la funció scanf?**

```
MacPepino:Desktop clara$ ./stringprova
Usuari, introduceix un string de llargada màxima 4
Sisplau sisplau fes-me cas que sinó no sé què passarà!
soc_dolent
Has introduït: soc_dolent
```

**Ai Déu meu, m'ha sobreescrit les variables següents... sort que no devien ser importants**

```
MacPepino:Desktop clara$ ./stringprova
Usuari, introduceix un string de llargada màxima 4
Sisplau sisplau fes-me cas que sinó no sé què passarà!
soc_unusuari_assassi_enserie
Has introduït: soc_unusuari_assassi_enserie
Segmentation fault: 11
```

**\$&%&\$\$!!! Ara sí que m'ha sobreescrit alguna cosa important per l'execució d'aquest programa...**

# Què passa si declaro un string massa petit?

**PREGUNTA LEGÍTIMA: I COM HO FAIG DONCS???**

# Què passa si declaro un string massa petit?

**PREGUNTA LEGÍTIMA: I COM HO FAIG DONCS???**

Calma! De moment en aquesta assignatura podem seguir fent servir **scanf**.



# Què passa si declaro un string massa petit?

## PREGUNTA LEGÍTIMA: I COM HO FAIG DONCS???

Calma! De moment en aquesta assignatura podem seguir fent servir **scanf**.

Però si ho volem saber, la solució seria fer servir alguna altra funció, com **fgets()**, que permet imposar una limitació del nombre de caràcters que llegim.

# Què passa si declaro un string massa petit?

## PREGUNTA LEGÍTIMA: I COM HO FAIG DONCS???

Calma! De moment en aquesta assignatura podem seguir fent servir **scanf**.

Però si ho volem saber, la solució seria fer servir alguna altra funció, com **fgets()**, que permet imposar una limitació del nombre de caràcters que llegim.

```
char *fgets( char *str, int count, FILE *stream );
```

[\[until C99\]](#)

```
char *fgets( char *restrict str, int count, FILE *restrict stream );
```

[\[since C99\]](#)

Reads at most `count - 1` characters from the given file stream and stores them in the character array pointed to by `str`. Parsing stops if a newline character is found, in which case `str` will contain that newline character, or if end-of-file occurs. If bytes are read and no errors occur, writes a null character at the position immediately after the last character written to `str`.

### Parameters:


- **str**: pointer to an element of a char array
- **count**: maximum number of characters to write (typically the length of `str`)
- **stream**: file stream to read the data from (Si volem que sigui entrada estàndard, `stdin`)

### Return value:

`str` on success, null pointer on failure.

# Què passa si declaro un string massa petit?

## Exemple amb fgets()



```
#include <stdio.h>
#define LLARG 5

int main(){

    char s[LLARG];
    printf("Usuari, introduceix un string de llargada màxima %d\n",LLARG-1);
    fgets(s, LLARG, stdin);
    printf("Has introduït: %s\n",s);
    return 0;
}
```

```
MacPepino:Desktop clara$ ./stringprova
Usuari, introduceix un string de llargada màxima 4
abcdefghijkl
Has introduït: abcd
```

# Què passa si declaro un string massa petit?

No m'ho puc creure! I per què no hem fet servir `fgets()` en comptes de `scanf()` en lo que portem de curs???

# Què passa si declaro un string massa petit?

No m'ho puc creure! I per què no hem fet servir `fgets()` en comptes de `scanf()` en lo que portem de curs???

**MOTIU 1: `scanf()`** és una lectura amb format. Això vol dir que posant `%d`, `%s`, `%c`, etc hem pogut llegir directament nombres, strings, caràcters... I `scanf` ens ha fet la conversió. Amb `fgets` només podem llegir strings, i llavors, si ho volem convertir a número, ho hauríem d'haver fet a mà.

# Què passa si declaro un string massa petit?

No m'ho puc creure! I per què no hem fet servir `fgets()` en comptes de `scanf()` en lo que portem de curs???

**MOTIU 1: `scanf()`** és una lectura amb format. Això vol dir que posant `%d`, `%s`, `%c`, etc hem pogut llegir directament nombres, strings, caràcters... I `scanf` ens ha fet la conversió. Amb `fgets` només podem llegir strings, i llavors, si ho volem convertir a número, ho hauríem d'haver fet a mà.

**MOTIU 2: `fgets()`** funciona diferent segons l'estàndard de C que facis servir per compilar:

```
char *fgets( char          *str, int count, FILE          *stream );      [until C99]
```

```
char *fgets( char *restrict str, int count, FILE *restrict stream );      [since C99]
```

# Què passa si declaro un string massa petit?


No m'ho puc creure! I per què no hem fet servir `fgets()` en comptes de `scanf()` en lo que portem de curs???

**MOTIU 3:** Si llegim amb `fgets` dues vegades, i a la primera posem més caràcters dels que toquen, a la segona lectura encara estarem llegint lo anterior.

# Què passa si declaro un string massa petit?

No m'ho puc creure! I per què no hem fet servir `fgets()` en comptes de `scanf()` en lo que portem de curs???

**MOTIU 3:** Si llegim amb `fgets` dues vegades, i a la primera posem més caràcters dels que toquen, a la segona lectura encara estarem llegint lo anterior.



```
#include <stdio.h>
#define LLARG 5

int main(){

    char s[LLARG];
    printf("Usuari, introdueix un string "
           "de llargada màxima %d\n",LLARG-1);
    fgets(s, LLARG, stdin);
    printf("Has introduit: %s\n",s);
    fgets(s, LLARG, stdin);
    printf("Has introduit: %s\n",s);
    return 0;
}
```



# Què passa si declaro un string massa petit?

No m'ho puc creure! I per què no hem fet servir `fgets()` en comptes de `scanf()` en lo que portem de curs???

**MOTIU 3:** Si llegim amb `fgets` dues vegades, i a la primera posem més caràcters dels que toquen, a la segona lectura encara estarem llegint lo anterior.

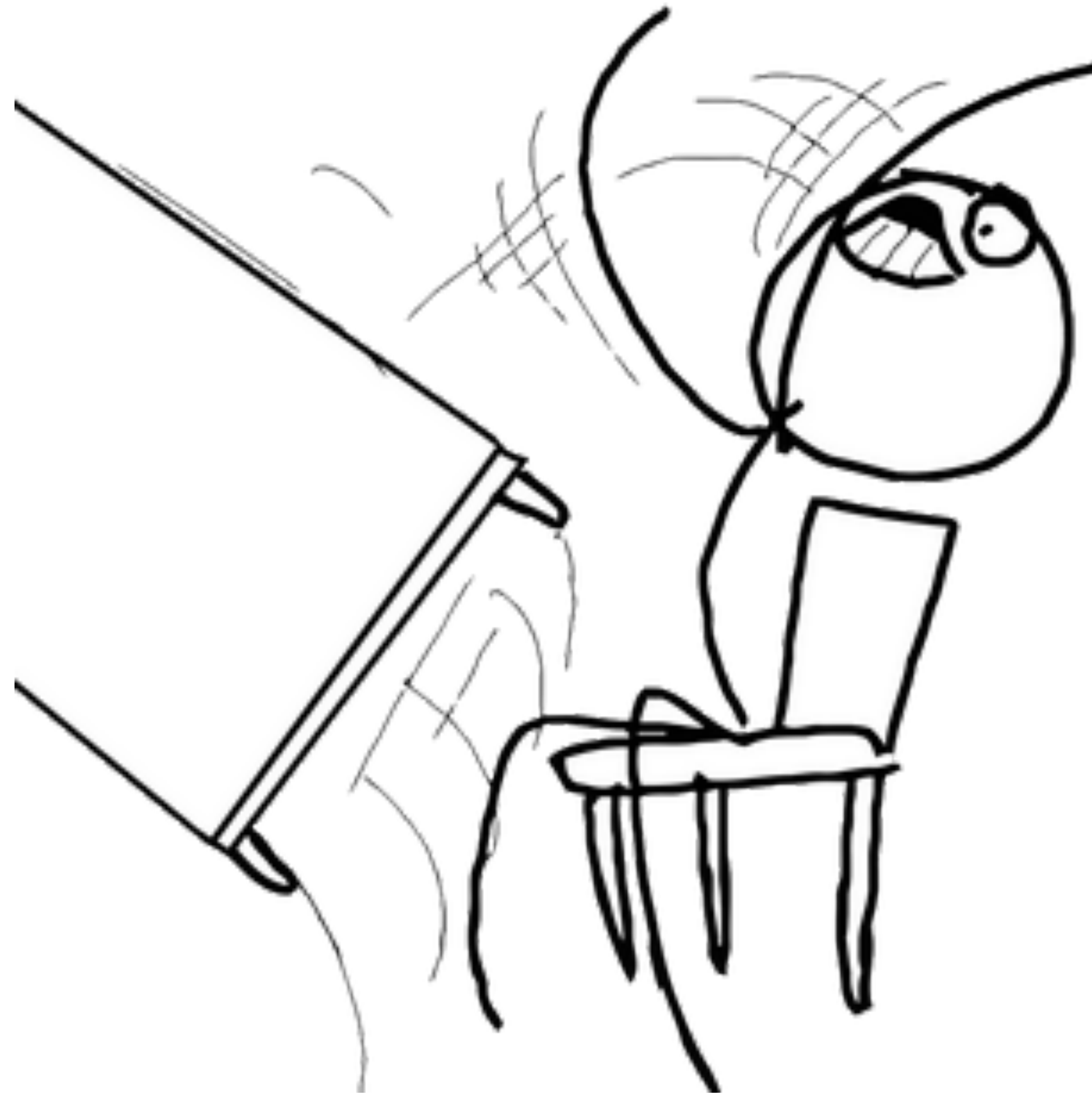
```
#include <stdio.h>
#define LLARG 5

int main(){

    char s[LLARG];
    printf("Usuari, introdueix un string "
           "de llargada màxima %d\n",LLARG-1);
    fgets(s, LLARG, stdin);
    printf("Has introduit: %s\n",s);
    fgets(s, LLARG, stdin);
    printf("Has introduit: %s\n",s);
    return 0;
}
```

```
Usuari, introdueix un string de llargada màxima 4
abcdefg
Has introduit: abcd
Has introduit: efg
```

**Què passa si declaro un string massa petit?**



## Bonus tip: Com trencar **strings** massa llargs al codi

```

#include <stdio.h>
#define LLARG 500

int main(){
    int i;
    char s[LLARG] = "Aquesta frase es una mica llarga"
                    " i com que no m'agrada tenir el codi"
                    " desordenat, la trenco en diverses línies."
                    " Fixeu-vos que ho estic indentant també. "
                    " S'ha de ser polit en aquesta vida ";

    printf("%s\n", s);
    return 0;
}
```

Aquesta frase es una mica llarga i com que no m'agrada tenir el codi desordenat, la trenco en diverses línies. Fixeu-vos que ho estic indentant també. S'ha de ser polit en aquesta vida

## Bonus tip: Com trencar **printfs** massa llargs al codi

```
● ● ●

#include <stdio.h>

int main(){
    int maj = 2;
    int min = 4;
    int dig = 5;
    int esp = 3;

    printf("Mare meva quin rollo de printf que m'espera."
        " El nombre de majúscules és %d,"
        " el nombre de minúscules es %d,"
        " el nombre de dígitos és %d,"
        " i el nombre de caràcters especials és %d.\n"
        "Fixeu-vos que segueixo indentant perquè sóc molt neta\n",
        maj, min, dig, esp);
    return 0;
}
```

Mare meva quin rollo de printf que m'espera. El nombre de majúscules és 2, el nombre de minúscules es 4, el nombre de dígitos és 5, i el nombre de caràcters especials és 3. Fixeu-vos que segueixo indentant perquè sóc molt neta