



TABLOURI CU TIP DE DATE CHAR ÎN C

String

DECLARAREA

```
#include <stdio.h>
int main(){
    char ch1[] = "Probleme cu String";
    printf("%s\n", ch1);
    ch1[20] = "Probleme cu String";
    printf("%s\n", ch1);
    ch1[10] = "Probleme cu String";
    printf("%s\n", ch1);
    ch1[] = {'P','r','o','b','l','e','m','e',' ','c','u',' ','S','t','r','i','n','g'};
    printf("%s\n", ch1);

    return 0;
}
```



**Ce se va
afişa?**

CITIREA & AFIȘAREA

```
#include <stdio.h>
int main(){
    char ch[] = "Probleme cu String";
    printf("%s\n", ch);
    scanf("%s", ch); // input: Probleme cu String
    printf("%s\n", ch);
    return 0;
}
```



Ce se va
afișa?

CITIREA & AFIȘAREA

```
#include <stdio.h>
int main(){
    char str[20], ch;
    int i = 0;
    do {
        ch = getchar();
        str[i] = ch;
        ++i;
    } while (ch != '\n');
    str[i-1] = '\0';
    printf("%s\n", str);
    return 0;
}
```

// input: Probleme cu String



Ce se va
afișa?

CITIREA & AFIȘAREA

- `scanf("%s",str);`
- `printf("%s", str);`
- `gets(str);` - citește de la terminalul standard un șir de caractere terminat cu linie nouă (enter, `\n`); are ca parametru adresa zonei de memorie în care se introduc caracterele citite și va returna adresa de început a zonei de memorie. *//va citi toate caracterele introduse, indiferent de memoria alocată*
- `fgets(str, len_max, stdin);` - citește de la terminalul standard un șir de caractere terminat cu linie nouă (enter, `\n`), dacă lungimea lui este mai mică decât lungimea maximă admisă șirului, în caz contrar, va citi doar primele `n-1` caractere, unde `n` este lungimea maximă admisă. În cazul în care șirul citit va avea o lungime mai mică decât cea maximă, înaintea terminatorului de șir (`\0`) în zona de memorie va fi reținut caracterul new-line (`\n`).
- `puts(str);` - afișează la terminalul standard șirul de caractere din zona data ca parametru, până la caracterul terminator de șir (`\0`), în locul căruia va afișa caracterul sfârșit de linie; are ca parametru adresa zonei de memorie de unde începe afișarea caracterelor și va returna codul ultimului caracter (diferit de `\0`) din șirul de caractere afișat sau `-1` dacă a apărut o eroare.

CITIREA & AFIŞAREA

```
#include <stdio.h>
int main(){
    char str[10];
    gets(str);
    puts(str);
    fgets(str, 10, stdin);
    puts(str);
    return 0;
}
```



Ce se va
afişa?

// input: Probleme cu String

CONCATENAREA A 2 ȘIRURI

```
#include <stdio.h>
#include <string.h>
int main(){
    char str[20] = "\0", str1 [] = "Probleme", str2[] = " cu String";
    strcat(str, str1);
    puts(str);
    strcat(str, str2);
    puts(str);

    return 0;
}
```

CONCATENAREA A 2 ȘIRURI

```
#include <stdio.h>
#include <string.h>
int main(){
    char str[256] = "\0", str1[100], str2[100];
    unsigned int len;
        fgets(str1, 100, stdin);
        len = strlen(str1);    // calculează lungimea sirului str1
        strncat(str, str1, len-1);    // concatenează str cu str1
        fgets(str2, 100, stdin);
        strncat(str, str2, strlen(str2)-1);
        puts(str);
        len = strlen(str);
        printf("len_str: %u", len);
return 0;
}
```


NUMĂRĂ APARIȚIILE LUI "A"

```
#include <stdio.h>
#include <string.h>
int main(){
    char str[50], ch = 'A';
    unsigned int len, i, count = 0;
    fgets(str, 50, stdin);
    len = strlen(str);
    for (i = 0; i < len-1; ++i){
        if (str[i] == ch) count++;
    }
    printf("Caracterul %c se repeta de %u ori", ch, count);
    return 0;
}
```

EXISTENȚA LITERELOR MARI

```
#include <stdio.h>
#include <string.h>
int main(){
    char str[50];
    unsigned int len, i, count = 0;
    fgets(str, 50, stdin);
    len = strlen(str);
    for (i = 0; i < len-1; ++i){
        if ((str[i] >= 'A') && (str[i] <= 'Z')) count++; }
    if (count != 0) {printf("Sirul contine litere mari");}
    else {printf("Sirul nu contine litere mari");}
    return 0;
}
```

SUMA CIFRELOR UNUI NUMAR

```
#include <stdio.h>
#include <string.h>
int main(){
    char str[50];
    unsigned int len, i, sum = 0;
    fgets(str, 50, stdin);
    len = strlen(str);
    for (i = 0; i < len-1; ++i){
        if ((str[i] >= '0') && (str[i] <= '9')) sum += (str[i]-48);    }
    printf("Sum = %u", sum);
return 0;
}
```