

DECLARAREA

char nume_sir[nr. maxim de caractere];
ex. char cuvant[15]

INITIALIZARE

char cuvant[]="calcul-
ulator"; char cuvant[15]="c-
alculator";

Sirul cuvant va fi Sirul cuvant va fi
compus din literele compus din literele
calculator, calculator avand 5
nemaiaavand spatii spatii libere
libere

CITIREA

```
cout<<"nr. de caractere=";
```

```
cin>>n;
```

```
for(i=0; i<n; i++)
```

```
{ cout<<"cuvant["<i<"=";
```

```
cin>>cuvant[i];
```

```
}
```

sau:

```
cin>>sir_de_caractere;
```

Se citeste un sir de caractere pana la primul spatiu liber/alb.

ex. a="Acesta este un exemplu"

```
cin>>a; (va afisa doar "Acesta")
```

Citirea unui sir de caractere incluzand spatiile libere se va face cu:

```
cin.get(sir, nr_caractere, '\n');
```

```
cin.getline(sir, nr_caractere)-include si sfarsitul liniei.
```

AFISAREA

```
for(i=0; i<n; i++)
```

```
cout<<cuvant[i];
```

TIPUL CHAR*

O variabila de tip pointer la caracter este capabila sa retina adresa de memorie a unui caracter

FUNCTII CARE OPEREAZA CU SIRURI DE CARACTERE

strlen(sir);

returneaza lungimea unui sir(numarul de caractere)

strcpy(destinatie,sursa);

copiază sursa peste destinatie

strcat(destinatie,sursa);

adauga sursa la destinatie, copiază sursa la sfarsitul destinatiei, conectează cele 2 siruri

strncat(destinatie,sursa,nr);

adauga primele n caractere din sursa la destinatie

strchr(sir,caracter);

returneaza adresa subsirului de caractere incepand cu prima pozitie a caracterului, caută caracterul in sir

strstr(sir1, sir2);

returneaza incepand de la prima aparitie a subsirului sir2 in sirul sir1 sau 0 in cazul in care nu exista

strcmp(sir1,sir2);

compara 2 siruri, returneaza >0 daca sir1>sir2; =0 daca sir1=sir2; <0 daca sir1<sir2

strtok(sir, separator);

returneaza primul sir de caractere pana la separatorul gasit

Transformarea unui caracter

-din litera mare in litera mica:

```
sir[i]=sir[i]+32;
```

-din litera mica in litera mare

```
sir[i]=sir[i]-32;
```