**Laborator 5**

**Siruri de caractere**

**Declaratie: *char nume\_sir[nr\_elemente];***

Libaria specifica lucrului cu siruri de caractere este: ***string.h/cstring***

Orice sir de caractere se termina cu caracterul special: **nul - \0.**

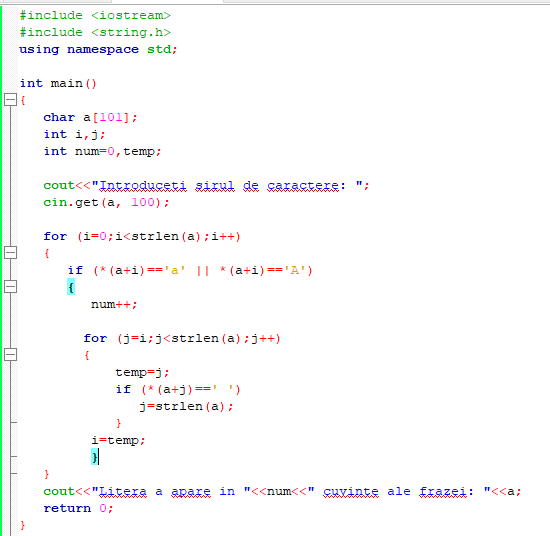
|  |  |  |
| --- | --- | --- |
| **Functie** | **Descriere** | **Exemplu** |
| char\* strcpy(char\*, char\*) | Copiaza valoarea reținută de o variabilă de tip char \* în altă variabilă de tip char \* | *char sir[40];*  *strcpy(sir1, “Buna ziua”);* |
| char\* strncpy(char\*, char\*, int) | Copiază primele n caractere din valoarea reținută de o variabilă de tip char \* în altă variabilă de tip char \* | *char sir1[30];*  *char sir2[] = ”Cristina Maria”;*  *strncpy(sir1, sir2, 7);* |
| int strlen(char \*) | Returnează numărul de caractere reținut de o variabilă char \* | *char \*sir = „Exemplu”;*  *printf(„%d\n”, strlen(sir));* |
| int puts(constr char\*) | Printeaza un sir de caractere | *puts(“Laboratorul 5”);* |
| char\* strcat (char\*, char\*) | Concateneaza 2 siruri de caractere | *char sir[100] =”Buna “;*  *char sir2[] = “ziua!”;*  *strcat(sir, sir2);* |
| char\* strncat (char\*, char\*, int) | Adauga la sfarsitul primului sir *len* caractere din cel de-al doilea sir | *char sir[100] =”Buna “;*  *char sir2[] = “ziua!”;*  *strcat(sir, sir2, 3);* |
| int strcmp (char\*, char\*) | Compara doua siruri, caracter cu caracter. Returneaza 0 daca cele doua siruri sunt egale si diferit de 0 altfel. | *char sir1[] = “Maria”;*  *char sir2[] = “Maria”;*  *int rezultat = strcmp(sir1, sir2);* |
| int strncmp (char\*, char\*, int) | Compara n caractere a doua siruri, caracter cu caracter. Returneaza 0 daca cele n caractere sunt egale si diferit de 0 altfel. | *char sir1[] = “Marius”;*  *char sir2[] = “Mariana”;*  *int rezultat = strcmp(sir1, sir2, 3);* |
| char \* strchr(char\* sir, char c) | Returneaza un pointer catre prima aparitie a caraterului in sir. | *char s[20] = “Limbaje”;*  *char \*p = strchr(s, ‘b’);*  *cout << p; //returneaza “baje”* |
| char \* strstr(char \*, char \*) | Returneaza un pointer catre prima aparitie a sirului 2 in sirul 1. | *char s[20] = “Limbaje de programare”;*  *char \*p = strchr(s, “de”);*  *cout << p; //returneaza “de programare”* |
| char strtok(char \* sir, char \* delim) | Se foloseste in principal pentru impartirea sirului cu ajutorul delimitatorilor | *char s[20] = “Limbaje de programare”;*  *char sep[] = “ .”;*  *char \*p = strtok(s, sep);*  *while(p != NULL)*  *{*  *cout <<p<<endl;*  *p = strtok(NULL, sep);*  *}* |

Functiile **toupper si tolower** convertesc un caracter la echivalentul sau in majuscule, respective litera mica.

*char rezultat = toupper(“c”); //returneaza C*

*char rezultat = tolower(“A”); //returneaza a*

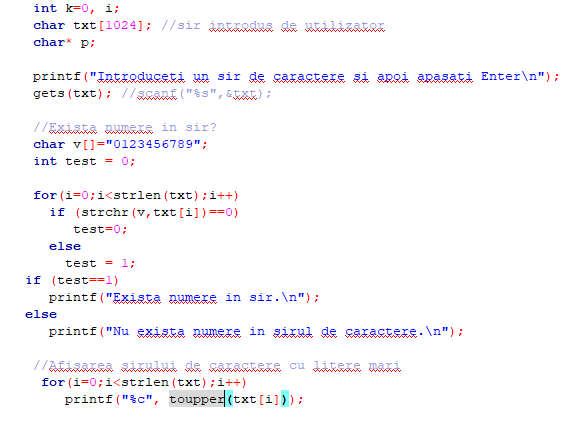
1. Sa se scrie un program care introduce de la tastatura o fraza si va afisa numarul de cuvinte care contin litera ‘a’ sau ‘A’.

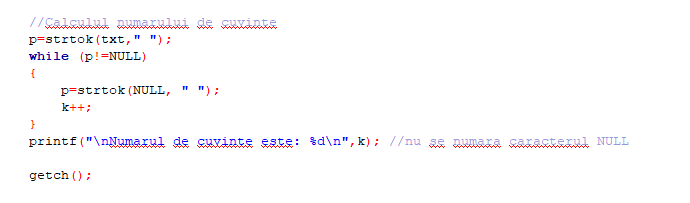


1. Se da un sir de caractere. Sa se afiseze:

* daca exista numere in sirul de caractere
* sirul de caractere cu litere mari
* numarul de cuvinte pe care acesta le contine

Rezolvare:





1. Se citesc de la tastatură doua siruri de maxim 100 de caractere. Concatenati cele doua siruri de caractere si impartiti sirul astfel format in subsiruri delimitate de spatiu sau “ .”.

Text

Description automatically generated

**Clasa string (doar pentru C++)**

O alta declaratie a sirurilor de caractere este cu ajutorul clasei **string.**

Initializarea si declararea unei clase string:

*string sir = “Exemplu”;*

*string sir(“Exemplu”);*

Citirea tipurilor de variabile **string**:

* **cin>>** aceasta varianta permite citirea stringurilor, cu exceptia spetiilor able (se foloseste in general pentru citirea unui singur cuvant)
* **getline(istream& is, string& str)** permite citirea strigurilor, inlcusiv a spetiilor able (ex: *getline(cin, sir))*

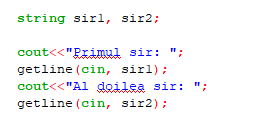
|  |  |  |
| --- | --- | --- |
| **Functie** | **Descriere** | **Exemplu** |
| + | Concatenarea a doua siruri de caractere | *string sir1 = “Tema”;*  *string sir2 = “Aleasa”;*  *string concat = si1 + sir2;* |
| int compare(string\* str1, string\* str2);  int compare(stirng& str); | Compararea a doua siruri de caractere | *string sir1 = “Alina”;*  *string sir2 = “Alina”;*  *int r = compare(si1, sir2);*  *int r = sir1.compare(sir2);* |
| string substr(string& str); | Returneaza un subsir al sirului | *string sir1 = sir2.substr(2, 6);* |
| int length(stirng& str);  int size(stirng& str); | Returneaza numarul de caractere din sir | *int n = sir1.length();*  *int n = sir1.size();* |
| int find(string& str); | Returneaza pozitia primului caracter din sirul cautat in sirul care se cauta, altfel -1 | *int pos = sir1.find(sir2);* |
| Metoda reserve() | Specifica dimensiunea unui string | *string sir(“Laborator”);*  *sir.reserve(15);* |
| Metoda clear() | Sterge continutul | sir.clear(); |
| Metoda append() | Adauga un sir la un alt sir | *string sir = “Exemplu ”;*  *string sir2 = “append”;*  *sir.append(sir2);*  *sir.append(sir2, 1, 3);* |
| Metoda push\_back() | Adauga un caracter la sfarsitului unui sir | *sir.push\_back(‘.’);* |
| Metoda insert() | Insereaza un sir la o pozitie specificata | *sir.insert(2, “mai”);* |

1. Se dau doua siruri de caractere de tipul “string”. Se cere:

* Lungimea fiecarui sir (*length(), size())*
* Daca cel de-al doilea sir este continut in primul sir de caractere (*find()*)
* Inserarea sirului “Limbaje de programare II” in sirul 1, incepand cu pozitia 3 (*insert()*)

Rezolvare:

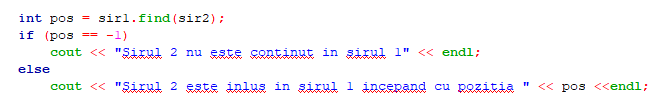
* Citirea sirurilor de caractere



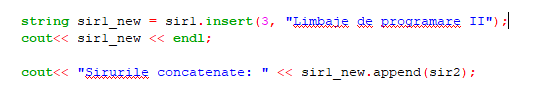
* Calcularea lungimii sirurilor cu ajutorul functiilor size() si length()



* Cautarea sirului 2 in sirul de caractere 1



* Inserarea sirului “Limbaje de programare II” pe pozitia 3 a primului sir si concatenarea celor 2 siruri



1. Se dau doua siruri de 100 de caractere. Sa se concateneze cele doua siruri, iar apoi din sirul nou format sa se extraga subsiruri delimitate prin spatiu sau punct.

Graphical user interface, text, application

Description automatically generated

1. Se dă o matrice cu n linii si m coloane, având toate elementele distincte. Să se calculeze:

* elementele de deasupra diagonalei principale (i<j)
* elementele de sub diagonala principala (i>j)
* elementele de deasupra diagonalei secundare (i+i<n-1 pentru i de la [0,n-1] si i+j<n+1 pentru i de la [1, n])
* elementele de sub diagonala secundara (i+i>n-1 pentru i de la [0,n-1] si i+j>n+1 pentru i de la [1, n])
* produsul elementelor de pe fiecare coloanal

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated