|  |
| --- |
| **Functii pentru caractere, siruri C si clase de sir C++** |

**Citirea caracerelor si a sirurilor de caractere**   
  
    Majoritatea calculatoarelor au instalat cel putin un editor de text. Functiile referitoare la siruri de caractere, respectiv la texte au deci o importanta capitala in implementarea diferitelor aplicatii de procesare texte. Sirurile de caractere sunt realizate din "insiruirea" mai multor caractere.   
    Vom incerca sa utilizam cateva functii referitoare la sirurile de caractere, disponibile in limbajul C++. Limbajul de programare C++, dispune de tipul de date *string*, cunoscut si sub numele de clasa sir C++.

 **Citirea unui caracter de la tastatura**   
  
    In capitolele anterioare am realizat o serie de aplicatii in care se astepta introducerea unui caracter sau a unui text.   
    Am utilizat astfel, la sfarsitul unui program, urmatoarea secventa de mai jos pentru a intreba utilizatorul daca vrea sa continue aplicatia sau renunta:   
    

|  |
| --- |
| // Citirea unui caracter de la tastatura  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Citirea unui caracter de la tastatura");// Titlul ferestrei consola  system("COLOR F9"); // Fundal alb caractere albastre  char car;  do{  cout << "\n\n\tApasati tasta E pentru iesire sau C pentru continuare :";  cin >> car;  if (car !='E' && car !='e')  cout << "\n\tAti ales -Continuare !";  else  cout << "\n\tAti ales -Iesire";  }while (car !='E' && car !='e');  return 0;  } |

    Dupa cum se observa, continuarea programului se face prin apasarea oricarei taste. Numai la apasara tastei E sau e se face iesirea din program. In cazul in care se apasa tasta Enter sau spatiu, nu se afiseaza nimic, functia cin << ignora caracterele spatiu si Enter de la inceput.   
    Pentru a lua in considerare si tasa Enter, vom folosi metoda **get** a obiectului **cin** . Metoda unui obiect se mai numeste si functia membru. In general, se utilizeaza expresia: "se invoca metoda **get** a obiectului **cin** .

|  |
| --- |
| // Citirea caracterului Enter  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Citirea caracterului 'Enter'");// Titlul ferestrei consola  system("COLOR F9"); // Fundal alb caractere albastre  char car;  do{  cout << "\n\n\tApasati tasta E pentru iesire sau Enter pentru continuare :";  cin.get(car);  if (car !='E' && car !='e')  cout << "\n\tAti ales -Continuare !";  else  cout << "\n\tAti ales -Iesire";  }while (car !='E' && car !='e');  return 0;  } |

    In cazul de sus continuarea aplicatiei se face apasand Enter. In cazul in care se introduce un caracter, urmat de Enter, programul se repeta de doua ori, adica de numarul de caractere introdus. Pentru a corecta aceasta problema, va trebui sa stergem bufferul de caractere dupa executia instructiunii **cin.get(car);** prin invocarea metodei **ignore** a obiectului **cin**. Trebuie avut insa grija sa nu stergem bufferul daca in el se afla numai caracterul Enter. Vom conditiona deci executia instructiunii : **cin.ignore();** cu o instructiune conditionala de genul **if (car !='\n')**

|  |
| --- |
| // Citirea caracterului Enter  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Citirea caracterului 'Enter'");// Titlul ferestrei consola  system("COLOR F9"); // Fundal alb caractere albastre  char car;  do{  cout << "\n\n\tApasati tasta E pentru iesire sau Enter pentru continuare :";  cin.get(car);  if (car !='\n')  cin.ignore();  if (car !='E' && car !='e')  cout << "\n\tAti ales -Continuare !";  else  cout << "\n\tAti ales -Iesire";  }while (car !='E' && car !='e');  return 0;  } |

 **Citirea unui sir de caractere**   
  
     Din aplicatiile anterioare s-a observat ca utilizand instructiunile:   
**string nume;   
    cin >> nume;**   
    nu putem citi numele si prenumele unei persoane pentru ca **cin** , va atribui variabilei nume, caracterele introduse pana la primul caracter spatiu, ignorand restul caracterelor. In cadrul tablourilor am rezolvat aceasta problema, invocand metoda **getline** a obiectului **cin** astfel:

|  |
| --- |
| // Citirea unui sir de caractere ce contine caractere "spatiu"  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Citirea unui sir de caracter de la tastatura");// Titlul ferestrei consola  system("COLOR F9"); // Fundal alb caractere albastre  char car;  char nume[80];  do{  cout << "\n\n\tIntroduceti numele d-voastra :";  cin.getline(nume,80);  cout << "\n\tNumele d-voastra este :" << nume ;  cout << "\n\n\tApasati tasta E pentru iesire sau Enter pentru continuare :";  cin.get(car);  if (car !='\n')  cin.ignore();  }while (car !='E' && car !='e');  return 0;  } |

**Functii pentru caractere si siruri de caractere**   
  
    In spatiul de nume std:: functiile pentru siruri sunt grupate in biblioteca "cstring" . Bibloteca "string" furnizeaza functii pentru calsa sir din C++.

 **Functii pentru caractere**   
  
     Exista situatii in care trebuie efectuam operatii asupra unui carater cum ar fi: sa testam valoarea unui caracter sau sa-l convertim din minuscula in majuscula etc. Exista functii special definite pentru operatii pe caractere.   
  
     **Functii pentru conversia tipului de caracter**   
  
    Sa reluam aplicatia de sus si sa folosim functia **touper** prin care sa convertim caracterul citit in litera mare astfel nu va mai fi necesara dubla testare (litera mare sau mica)

|  |
| --- |
| // Utilizrea functiei "toupper"  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Utilizrea functiei 'toupper'");// Titlul ferestrei consola  system("COLOR F9"); // Fundal alb caractere albastre  char car;  char nume[80];  do{  cout << "\n\n\tIntroduceti numele d-voastra :";  cin.getline(nume,80);  cout << "\n\tNumele d-voastra este :" << nume ;  cout << "\n\n\tApasati tasta E pentru iesire sau Enter pentru continuare :";  cin.get(car);  car=toupper(car);  if (car !='\n')  cin.ignore();  }while (car !='E');  return 0;  } |

     **Functii pentru verificarea valorii unui caracter**   
  
    De multe ori trebuie sa verificam valoarea unui caracter de exemplu: sa testam daca un caracter este o litera sau o cifra, sa testam o litera este majuscula sau minscula etc.   
  
    Vom realiza in continuare un program care cere o litera, o analizeaza si afiseaza tipul caracterului.

|  |
| --- |
| // Functii pentru verificarea valorii unui caracter  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Utilizrea functiilor pe caracter");// Titlul ferestrei consola  system("COLOR F9"); // Fundal alb caractere albastre  char car;  char car\_t;  do{  cout << "\n\n\tIntroduceti un caracter :";  cin.get(car\_t);  if (isalnum(car\_t)){  if (isalpha(car\_t)){  if (isupper(car\_t))  cout << "\n\n\tAti introdus un caracter majuscula :";  else  cout << "\n\n\tAti introdus un caracter minuscula :";  }else{  cout << "\n\n\tAti introdus o cifra :";  }  }else{  cout << "\n\n\tNu ati introdus nici cifra nici caracter :";  }  cin.ignore();  cout << "\n\n\tApasati tasta E pentru iesire sau Enter pentru continuare :";  cin.get(car);  car=toupper(car);  if (car !='\n')  cin.ignore();  }while (car !='E');  return 0;  } |

 **Functii pentru siruri de caractere**   
  
    **Determinarea lungimii unui sir**   
  
    Functia **strlen**   
  
    Forma generala: **strlen (nume\_sir)**   
    Functia determina lungimea unui sir. Functia returneaza un numar întreg ce reprezinta lungimea unui sir de caractere, fara a numara terminatorul de sir.   
    Urmatoarea aplicatie determina lungimea a trei siruri declarate in moduri diferite.

|  |
| --- |
| // Programul afiseaza lungimea unui sir de caractere  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void)  {  system("TITLE Determinarea lungimii unui sir ");  system("COLOR F9");  cout << "\n\t- Programul afiseaza lungimea diverselor siruri de caractere.\n\n\n";  char\* fc ="Facultatea de Inginerie";  cout <<"\n\n\tLungimea sirului: " << fc << " este: " << strlen(fc);  char ctd[55]="Inginerie Electrica";  cout <<"\n\n\tLungimea sirului: " << ctd << " este: " << strlen(ctd);  cout <<"\n\n\tLungimea sirului: " << "Sectia Calculatoare" << " este: " << strlen("Sectia Calculatoare");  cin.get();  return 0;  } |

    Functia **strlen** poate fi folosita de asemenea pentru a determina lungimea unui sir preluat de la utilizator.

|  |
| --- |
| // Programul afiseaza lungimea unui sir de caractere introdus de la tastatura.  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void)  {  system("TITLE Determinarea lungimii unui sir preluat ");  system("COLOR F9");  cout << "\n\t- Programul afiseaza lungimea unui sir preluat de la tastatura.\n\n\n";  char n\_pr[80];  cout << "\n\t- Introduceti numele d-voastra:";  cin.getline(n\_pr,80);  cout <<"\n\n\tLungimea sirului: " << n\_pr << " este: " << strlen(n\_pr);  cin.ignore();  cin.get();  return 0;  } |

    Sa combinam functiile pe caractere cu functiile pe siruri pentru a verifica daca un sir este introdus corect. Sa luam de exemplu o adresa de mail. Vom verifica daca are cel putin 5 caractere si contine caracterele . @.

|  |
| --- |
| // Programul cere adresa de mail si verifica daca:  // - adresa are cel putin 5 caractere  // - adresa contine caracterul .  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void)  {  system("TITLE Validarea unui sir preluat ");  system("COLOR F9");  bool dim=false, at=false, punct=false;  cout << "\n\tProgramul cere adresa de mail si verifica daca:.\n";  cout << "\n\t - adresa are cel putin 5 caractere";  cout << "\n\t - adresa contine caracterul @";  cout << "\n\t - adresa contine caracterul .";  char adresa[80];  cout << "\n\n\n\t- Introduceti adresa de mail:";  cin.getline(adresa,80);  if (strlen(adresa) >5){  dim=true;  }  for (int i=0; i <= strlen(adresa); i++){  if (adresa[i]=='@')  at=true;  if (adresa[i]=='.')  punct=true;  }  if(((at)&(punct)&(dim))==0){  if (dim==0)  cout <<"\n\n\tLungimea adresei: " << adresa << " este este mai mica de 5 caractere!";  if (at==0)  cout <<"\n\n\tAdresa : "<< adresa << "nu contine @ ";  if (punct==0)  cout <<"\n\n\tAdresa : "<< adresa << " nu contine .";  }else{  cout <<"\n\n\tAdresa : "<< adresa << " este corecta !";  }  cin.get();  return 0;  } |

    Operatia efectuata anterior se numeste operatia de validare camp.   
  
    **Copierea unui sir intr-un alt sir**   
  
    Forma generala: **strcpy (sir\_destinatie, sir\_sursa)**   
    Functia copiaza sirul sursa în sirul destinatie. Pentru a fi posibila copierea, lungimea sirului destinatie trebuie sa fie mai mare sau egala cu cea a sirului sursa, altfel pot apare erori.   
    Sa luam o aplicatie de geniul:

|  |
| --- |
| //Copierea unui sir intr-un alt sir  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void)  {  system("TITLE Copierea unui sir intr-un alt sir ");  system("COLOR F9");  char\* sir1="Alfa";  char\* sir2="Beta";  sir1=sir2;  cout << "\n\n\tSirul 1 este: " << sir1;  cin.get();  return 0;  } |

    Aplicatia nu cpiaza continutul sir 2 peste sir 1 ci atribuie pointerului sir1 valoarea pointerului sir2.   
    Pentru a demonstra acest lucru, vom mai introduce un pointer numit sir\_1 care va indica tot spre sir1

|  |
| --- |
| //Copierea unui sir intr-un alt sir  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  char\* sir\_1;  int main(void){  system("TITLE Copierea unui sir intr-un alt sir ");  system("COLOR F9");  char\* sir1="Alfa";  char\* sir2="Beta";  sir\_1=&sir1[0];// pointerul sir\_1 indica spre sir1  sir1=sir2;  cout << "\n\n\tSirul 1 este: " << sir1;  // daca in urma operatiei sir1=sir2 s-ar copia sir2 peste sir1, instructiunea de jos  // ar trebui sa afiseze tot "Beta" insa afiseaza "Alfa"  cout << "\n\n\tSirul 1 este: " << sir\_1;  cin.get();  return 0;  } |

    Pentru a copia efectiv sirul, vom folosi functia **strcpy** :

|  |
| --- |
| //Copierea unui sir intr-un alt sir  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  char\* sir\_1;  int main(void){  system("TITLE Copierea unui sir intr-un alt sir ");  system("COLOR F9");  char sir1[5]="Alfa";  char sir2[5]="Beta";  sir\_1=&sir1[0];// pointerul sir\_1 indica spre sir1  strcpy(sir1,sir2);  cout << "\n\n\tSirul 1 este: " << sir1;  cout << "\n\n\tSirul 1 este: " << sir\_1;  cin.get();  return 0;  } |

    De data aceasta se afiseaza acelasi lucru deci copierea s-a facut efectiv.   
    In cazul ca vrem sa copiem un numar limitat de caractere, folosim functia **strncpy** astfel:

|  |
| --- |
| //Copierea unui subsir intr-un alt sir  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void){  system("TITLE Copierea unui sir intr-un alt sir ");  system("COLOR F9");  char sir1[10]="Alfa-Soft";  char sir2[5]="Beta";  strncpy(sir1,sir2,3);  cout << "\n\n\tSirul 1 este: " << sir1;  cin.get();  return 0;  } |

    Raspunsul aplicatiei va fi de data aceasta: "Beta-Soft"   
  
    **Concatenarea a doua siruri**   
  
    Forma generala: **strcat (sir\_destinatie, sir\_sursa)**   
    Operatia de adunare a doua siruri adica operatia de a copia un sir la sfarsitul altui sir se numeste concatenare.   
    Functia **strcat** concateneaza cele doua siruri: sirul sursa este adaugat la sfârsitul sirului destinatie. Tabloul care contine sirul destinatie trebuie sa aiba cel putin dimensiunea celor doua siruri.   
    Pentru a concatena doua siruri, procedam astfel:

|  |
| --- |
| //Adaugarea unui sir la un alt sir  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void){  system("TITLE Adaugarea unui sir la un alt sir ");  system("COLOR F9");  char sir1[15]="Alfa-Soft-";  char sir2[5]="Beta";  strcat(sir1,sir2);  cout << "\n\n\tSirul 1 este: " << sir1;  cin.get();  return 0;  } |

    Raspunsul aplicatiei va fi de data aceasta: "Alfa-Soft-Beta". Trebuie sa avem grija ca dimensiunea sirului sir1 sa fie suficienta pentru a adauga sir2.   
  
    **Compararea a doua siruri**   
  
    Forma generala: **strcmp (sir\_1, sir\_2)**   
    Functia compara cele doua siruri date ca argument si returneaza o valoare întreaga egala diferenta dintre codurile ASCII ale primelor caractere care nu coincid.   
    Pentru a compara doua siruri, vom folosi functia **strcmp** . Functia returneaza 0 daca sirurile sunt egale, un numar negativ daca primul sir< al doilea sir respectiv pozitiv daca primul sir > al doilea sir.

|  |
| --- |
| //Compararea a doua siruri;  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void){  system("TITLE Compararea a doua siruri ");  system("COLOR F9");  char sir1[10]="Alfa";  char sir2[5]="Alfa";  if (strcmp(sir1,sir2)==0)  cout << "\n\n\tCele doua siruri sunt egale !" ;  else  cout << "\n\n\tCele doua siruri sunt diferite !" ;  cin.get();  return 0;  } |

    Raspunsul este desigur "Cele doua siruri sunt egale". Daca am folosi conditia :"if (sir1==sir2) raspunsul ar fi tot timpul fals deoarece se compara defapt doua adrese ale celor doua siruri.   
  
    **Conversia unui intreg intr-un sir**   
  
    

|  |
| --- |
| // Conversia unui intreg intr-un sir  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Conversia unui intreg intr-un sir");  system("COLOR F9");  int n;  char buffer [33];  cout << "\n\n\tIntroduceti un numar intreg: " ;  cin >> n;  itoa(n,buffer,10);//functia pentru conversia unui intreg intr-un sir  cout << "\n\n\tSirul rezultat din convertirea numarului este: " << buffer;  cin.ignore();  cin.get();  return 0;  } |

    **Pozitia unde apare subsirul s2 in sirul s1**   
  
    Forma generala: **strstr(s1, s2)**   
    Functia returneaza pozitia unde apare subsirul s2 in sirul s1 sau altfel spus, functia returneaza un pointer la începutul subsirului s2, sau NULL daca subsirul nu este gasit.   
    Urmatoarea aplicatie foloseste functia *strstr(s1, s2)* pentru a gasi pozitia unde se gaseste un subsir si afiseaza sirul initial incepand din aceasta pozitie.

|  |
| --- |
| // Programul cere un sir si un subsir de caractere .  // Programul afiseaza sirul din pozitia unde incepe subsirul in cadrul sirului .  // Se foloseste functia  // Functia strstr(s1, s2) – returneaza pozitia unde apare subsirul s2 in sirul s1  #include "stdafx.h"  #include < iostream >  #include < string >  using namespace std;  int main(void)  {  system("TITLE Pozitia unui subsir in cadrul unui sir ");  system("COLOR F9");  char s1[100],s2[25],\*p;  cout << "\n\n\tIntroduceti sirul = ";  cin.getline(s1,80);  cout << "\n\tIntroduceti subsirul = ";  cin.getline(s2,80);  p=strstr(s1,s2);  if (p!=NULL)  cout << "\n\n\n\tSirul este: = " << p;  else  cout << "\n\n\n\tSubsirul: " << s2 << " nu s-a gasit";  cin.ignore();  cin.get();  return 0;  } |

    **Conversia unui sir intr-un intreg**   
  
    Pentru a converti un text ce reprezinta un intreg folosim functia **atoi**

|  |
| --- |
| // Conversia unui sir intr-un intreg  #include "stdafx.h"  #include < iostream >  using namespace std;  int main(void)  {  system("TITLE Conversia unui sir intr-un intreg ");  system("COLOR F9");  char nr\_ascii [10]="123";  int nr=atoi(nr\_ascii);  cout << "\n\n\tValoarea zecimala a sirului este: " << nr ;  cout << "\n\n\tPatratul valoarii zecimale a sirului este: " << nr\*nr;  cin.ignore();  cin.get();  return 0;  } |

**Siruri de caracere in spatiul System**   
  
    **Diverse conversii**   
  
    In spatiul de nume System se foloseste System::Convert::... pentru diverse conversii.

|  |
| --- |
| // Conversia unui sir intr-o valoare numerica in System  // Functia ReadLine citeste tot timpul un sir de caractere  // Se foloseste System::Convert::... pentru diverse conversii  #include "stdafx.h"  #include < iostream >  using namespace std;  using namespace System;  int main(void)  {  std::system("TITLE Conversia unui sir intr-o valoare numerica");  std::system("COLOR F9");  double raza; // valoarea numerica a razei  String^ raza\_s; // sirul ce contine valoarea citita  double pi= System::Math::PI;  Console::Write( L"\n\n\tIntroduceti raza cercului:" );  raza\_s= Console::ReadLine();  raza = System::Convert::ToDouble( raza\_s );  Console::WriteLine( "\n\n\tAria cercului de raza: "+ raza +" este: "+pi\*raza\*raza );  Console::ReadLine();  return 0;  } |

    La fel procedam si pemtru aplicatii CLR Windows Form Applications.   
    Pentru a realiza o aplicatie care permite sa zicem introducerea unei valori numerice care reprezinta un unghi in radiani dupa care afiseaza sinusul acelei valori, avem nevoie de o functie ce converteste valoarea in Double, pentru a calcula sinusul apoi avem nevoie de conversia valorii sinusului intr-un text in vederea afisarii intr-un obiect de tip label.   
    Deschidem un nou proiect Windows Forms Application intitulat "convers\_01" pe care plasam un obiect de tip numericUpDown numit numericUpDown1 si doua obiecte de tip label numite label1,label 2.   
  
    Completam procedura deschisa pe evenimentul "ValueChanged" cu:

|  |
| --- |
| double rad2;  rad2=System::Convert::ToDouble(this->numericUpDown1->Value);  this->label2->Text =System::Convert::ToString(System::Math::Sin(rad2)); |

    C#

|  |
| --- |
| double rad2;  rad2=System.Convert.ToDouble(this.numericUpDown1.Value);  this.label2.Text =System.Convert.ToString(System.Math.Sin(rad2)); |

    Metoda: **System.Convert.ToDouble(this.numericUpDown1.Value)**- converteste valoarea introdusa in Double iar **System.Convert.ToString(System.Math.Sin(rad2))** converteste vaoarea double in text.   
    Dupa rularea aplicatiei, tastarea unei valori, apoi tasta Enter, se obtine:

    Obiectul numericUpDown poate fi inlocuit cu un obiect de tipul textBox numit textBox1. Avantajul este ca dupa fiecare tastare a unei cifre se afiseaza o noua valoare a sinusului. Dezavantajul este ca daca se tasteaza alte caractere inafara cifrelor, programul da o eroare de exceptie care trebuie tratata. In acest caz procedura deschisa pe evenimentul "ValueChanged" a obiectului textBox1 devine:

|  |
| --- |
| double rad2;  try {  rad2=System::Convert::ToDouble(this->textBox1->Text );  this->label2->Text =System::Convert::ToString(System::Math::Sin(rad2));  }  catch(System::FormatException^){  this->label2->Text="Eroare format";  } |

    C#

|  |
| --- |
| double rad2;  try {  rad2=System.Convert.ToDouble(this.textBox1.Text );  this.label2.Text=System.Convert.ToString(System.Math.Sin(rad2));  }  catch(System.FormatException){  this.label2.Text="Eroare format";  } |

     **Metode pentru siruri utilizate in WFA**   
  
    Vom utiliza in continuare cateva metode referitoare la siruri de caractere. Vom realiza o aplicatie WFA care afiseaza in mod continuu numarul de caractere introduse de la tastatura. In cadrul acestei aplicatii vom utiliza un obiect label care ne va permite sa afisam numarul de caractere introdus de la tastatura prin intermediul unui obiect de tip TextBox. Vom utiliza un obiect ProgressBar care ne va permite sa afisam grafic procentul de caractere introdus. Afisarea se va face procentual relativ la numarul maxim de 300 de caractere setat pentru aceasta aplicatie. Se va afisa de asemenea si procentul sub forma numerica, atat pe 15 caractere cat si pe maxim 5 caractere.   
    Vor fi folosite printre altele metodele :

* **Length** -pentru determinarea lungimii unui sir de caractere
* **Trim** -pentru eliminarea spatiilor din cadrul unui sir de caractere
* **System::String::Substring** - pentru a substrage un subsir dintr-un sir de caractere
* **System::Convert::ToString** - pentru a converti diverse tipuri in string

    Deschidem un nou proiect Windows Forms Application intitulat "text\_bar" pe care plasam:

* 7 obiecte de tip **label**
* un obiect de tip **textBox** caruia ii setam proprietatea "multiline" la "True".
* un obiect de tip **progressBar**
* un obiect de tip **timer** caruia ii setam proprietatea "enabled" la "True" si interval la 100.

    Completam procedura deschisa pe evenimentul Tick al obiectului timer1 cu :

|  |
| --- |
| double nr\_car=(this->textBox1->Text)->Length;  this->label1->Text=System::Convert::ToString(nr\_car);  double proc=System::Convert::ToDouble(100\*nr\_car/300);  String^ proc\_s=System::Convert::ToString(proc);  this->label2->Text=proc\_s;  String^ proc\_st=proc\_s->Trim();  String^ proc\_s5;  if (proc\_st->Length > 5)  proc\_s5=proc\_s->System::String::Substring(0,5);  else  proc\_s5=proc\_s;  if (proc>100)  proc=100;  this->label3->Text=proc\_s5;  this->progressBar1->Value =proc; |

    C#:

|  |
| --- |
| namespace text\_bar  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  private void timer1\_Tick(object sender, EventArgs e)  {  String proc\_s, proc\_st, proc\_s5;  double nr\_car=(this.textBox1.Text).Length;  double proc;  this.label2.Text=System.Convert.ToString(nr\_car);  proc=System.Convert.ToDouble(100\*nr\_car/300);  proc\_s=System.Convert.ToString(proc);  this.label4.Text=proc\_s;  proc\_st=proc\_s.Trim();  if (proc\_st.Length > 5)  proc\_s5=proc\_s.Substring(0,5);  else  proc\_s5=proc\_s;  if (proc>100)  proc=100;  this.label6.Text=proc\_s5;  this.progressBar1.Value = System.Convert.ToInt16(proc);  }  }  } |

    Rulam aplicatia si obtinem:

**Vehicularea sirurilor de caractere prin intermediul portului serial**  
    Portul serial a fost utilizat pentru a realiza prima legatura intre doua calculatoare. Portul serial este des utilizat si in prezent pentru a conecta diverse dispozitive la calculator. Chiar daca majoritatea calculatoarelor nu mai dispun fizic de un port serial, se utilizeaza des porturi seriale virtuale, realizate prin intermediul portului USB (Universal Serial Bus). Cu alte cuvinte exista o serie de dispozitive conectate la calculator prin intermediul USB insa din punct de vedere logic ele sunt conectate printr-un port serial virtual.   
    Vom lucra cu un sistem de achizitie date realizat pe baza unui sistem de dezvoltare PIC   
    Sistemul de achizitie dispune de 5 intrari analogice 4 digitale si 8 iesiri digitale, conectate astfel:

* Comanda pentru citirea simultana a celor cinci canale analogice este:**Ai**
* Comanda pentru citirea unei intrari analogice este:**Ai** unde i=0..4,
* Comanda pentru inscrierea celor 9 iesiri digitale este **On** unde n este numarul in zecimal care va fi afisat in binar pe cele 8 iesiri digitale
* Comanda D1 pentru citirea unui numar zecimal care codifica cele 4 intrari digitale.

     Pentru programator, conectarea prin USB este identica cu o conectare prin intermediul unui port serial fizic. Conectarea prin intermediul portului USB fiind deci "transparenta" din punctul de vedere al scrierii aplicatiilor, acestea fiind scrise la fel cu aplicatiile pentru portul serial.

 **Configurarea si initializarea portului serial**   
  
**C# - selectarea portului dorit**

|  |
| --- |
| int i, j;  // Listez porturile seriale  Nume\_porturi = System.IO.Ports.SerialPort.GetPortNames();  this.Porturi\_s.Items.Clear();  //Adaug porturile exixtente  for (i = 0; i < Nume\_porturi.Length; i++)  {  this.Porturi\_s.Items.Add(Nume\_porturi[i]);  }  //Pozitionarea listei pe primul element  this.Porturi\_s.SelectedIndex = 0; |

     Unde "Porturi\_s" este un obiect de tip ListBox.   
  
**C# - Conectarea la portul dorit**

|  |
| --- |
| if (!this.serialPort1.IsOpen)  {  this.serialPort1.PortName = System.Convert.ToString(this.Porturi\_s.Items[this.Porturi\_s.SelectedIndex]);  this.serialPort1.Open();  this.label1.Text = "Portul a fost deschis";  }  else  {  this.label1.Text = "Portul este deja deschis deschis";  }  this.textBox1.Text = "AA";  this.serialPort1.Write("AA"); |

**C# - Deconectarea de la portul serial**

|  |
| --- |
| this.serialPort1.Close();  this.label1.Text = "Portul este deconectat!"; |

**C# - scrierea si citirea portului serial**

|  |
| --- |
| cda = this.textBox1.Text;    if (this.serialPort1.IsOpen)  {  this.serialPort1.Write(cda);  txt = "";  txt = this.serialPort1.ReadExisting();  if (txt.Length > 0)  {  this.textBox2.Text = txt;  }  else  {  this.label1.Text = "Nu vin date!";  }  } |

**Utilizare sistem de achizitie**   
  
    Pentru a putea fi utilizat, portul serial trebuie configurat, cu alte cuvinte trebuiesc stabilite valorile pentru diversi parametri, cum ar fi: viteza de lucru, lungimea cuvantului, paritate etc.   
    In vederea utilizarii portului serial, deschidem un nou proiect C# Windows Forms Application intitulat "RS\_232\_v1" . In cadrul acestei aplicatii, e nevoie de o interfata care sa permita selectarea portului serial, deschiderea si inchiderea acestuia, stabilirea unui sir de caractere pentru a fi trimis prin portul serial, trimiterea acestuia, citirea unui sir de caractere si afisarea sirului citit. Vom plasa deci

* 2 obiecte de tip **button** 2 pentru conectarea si deconectarea la portul serial(cu numele "but\_con" respectiv "but\_dec"),
* un obiect de tip **ListBox** cu numele "Porturi\_s" pentru alegerea portului serial
* 2 obiecte de tip **textBox** pentru comanda respectiv pentru afisarea raspunsului
* un obiect de tip **timer**
* un obiect de tip **serialPort**

    Obiectul de tip "ListBox" este folosit pentru selectarea unui port serial. La un moment dat pot fi deschise mai multe porturi seriale, deci trebuie sa dispunem de o metoda prin care sa selectam portul serial dorit.   
    Obiectul de tip "ListBox" este cel mai potrivit pentru a selecta unul din porturile seriale cu conditia ca in momentul lansarii aplicatiei sa fie insctrise elementele listei cu numele porturilor deschise in acel moment.

 **Citire intrari analogice**

    Codul sursa fiind:

|  |
| --- |
| namespace RS\_232\_v01  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  string txt, cda;  int k;  static String[] Nume\_porturi = new String[11];  private void Form1\_Load(object sender, EventArgs e)  {  int i, j;  // Listez porturile seriale  Nume\_porturi = System.IO.Ports.SerialPort.GetPortNames();  this.Porturi\_s.Items.Clear();  //Adaug porturile exixtente  for (i = 0; i < Nume\_porturi.Length; i++)  {  this.Porturi\_s.Items.Add(Nume\_porturi[i]);  }  //Pozitionarea listei pe primul element  this.Porturi\_s.SelectedIndex = 0;  this.label1.Text = "Conectati-va la portul serial!";  }  private void but\_con\_Click(object sender, EventArgs e)  {  if (!this.serialPort1.IsOpen)  {  this.serialPort1.PortName = System.Convert.ToString(this.Porturi\_s.Items[this.Porturi\_s.SelectedIndex]);  this.serialPort1.Open();  this.label1.Text = "Portul a fost deschis";  }  else  {  this.label1.Text = "Portul este deja deschis deschis";  }  this.textBox1.Text = "AA";  this.serialPort1.Write("AA");  }  private void but\_dec\_Click(object sender, EventArgs e)  {  this.serialPort1.Close();  this.label1.Text = "Portul este deconectat!";  }  private void timer1\_Tick(object sender, EventArgs e)  {  k++;  if (k > 123456789)  k = 0;  this.label2.Text = k.ToString();  cda = this.textBox1.Text;    if (this.serialPort1.IsOpen)  {  this.serialPort1.Write(cda);  txt = "";  txt = this.serialPort1.ReadExisting();  if (txt.Length > 0)  {  this.textBox2.Text = txt;  }  else  {  this.label1.Text = "Nu vin date!";  }  }  }  }  } |

    Datele primite trebiesc despachetate pentru a putea fi utilizate. Urmatoarea aplicatie "RS\_232\_v02" Extrage valorie V4-V0 din sirul primit.

|  |
| --- |
| namespace RS232\_v02  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  string cda, rec;  int k;  static String[] Nume\_porturi = new String[11];  private void Form1\_Load(object sender, EventArgs e)  {  int i, j;  // Listez porturile seriale  Nume\_porturi = System.IO.Ports.SerialPort.GetPortNames();  this.Porturi\_s.Items.Clear();  //Adaug porturile exixtente  for (i = 0; i < Nume\_porturi.Length; i++)  {  this.Porturi\_s.Items.Add(Nume\_porturi[i]);  }  //Pozitionarea listei pe primul element  this.Porturi\_s.SelectedIndex = 0;  this.label1.Text = "Conectati-va la portul serial!";  }  private void but\_con\_Click(object sender, EventArgs e)  {  if (!this.serialPort1.IsOpen)  {  this.serialPort1.PortName = System.Convert.ToString(this.Porturi\_s.Items[this.Porturi\_s.SelectedIndex]);  this.serialPort1.Open();  this.label1.Text = "Portul a fost deschis";  }  else  {  this.label1.Text = "Portul este deja deschis deschis";  }  this.textBox1.Text = "AA";  this.serialPort1.Write("AA");  }  private void but\_con\_Click\_1(object sender, EventArgs e)  {  if (!this.serialPort1.IsOpen)  {  this.serialPort1.PortName = System.Convert.ToString(this.Porturi\_s.Items[this.Porturi\_s.SelectedIndex]);  this.serialPort1.Open();  this.label1.Text = "Portul a fost deschis";  }  else  {  this.label1.Text = "Portul este deja deschis deschis";  }  this.textBox1.Text = "AA";  this.serialPort1.Write("AA");  }  private void but\_dec\_Click(object sender, EventArgs e)  {  this.serialPort1.Close();  this.label1.Text = "Portul este deconectat!";  }  private void timer1\_Tick(object sender, EventArgs e)  {  k++;  if (k > 123456789)  k = 0;  this.label2.Text = k.ToString();  cda = this.textBox1.Text;  if (this.serialPort1.IsOpen)  {  this.serialPort1.Write(cda);  rec = "";  rec = this.serialPort1.ReadExisting();  if (rec.Length > 0)  {  this.textBox2.Text = rec;  int p1 = rec.IndexOf(",");  int p2 = rec.IndexOf(",", p1 + 1);  int p3 = rec.IndexOf(",", p2 + 1);  int p4 = rec.IndexOf(",", p3 + 1);  int p5 = rec.IndexOf(",", p4 + 1);  int p6 = rec.IndexOf(",", p5 + 1);  this.label7.Text = "V4: \n\n" + rec.Substring(p1 + 1, p2 - p1 - 1);  his.label8.Text = "V3: \n\n" + rec.Substring(p2 + 1, p3 - p2 - 1);  this.label9.Text = "V2: \n\n" + rec.Substring(p3 + 1, p4 - p3 - 1);  this.label10.Text = "V1: \n\n" + rec.Substring(p4 + 1, p5 - p4 - 1);  this.label11.Text = "V0: \n\n" + rec.Substring(p5 + 1, p6 - p5 - 1);  }  else  {  this.label1.Text = "Nu vin date!";  this.label10.Text = " ";  this.label9.Text = " ";  this.label8.Text = " ";  this.label7.Text = " ";  }  }  }  }  } |

    Se observa ca p1-p6 sunt pozitiile in care se gaseste separatorul "," iar p6 este pozitia in care se termina sirul, respectiv lungimea acestuia. Pozitia este determinata folosind metoda "IndexOf". Pentru a extrage parametrul curent, stiindu-se pozitia celor doi separatori "," adica inceputul si sfarsitul subsirului ce reprezinta parametrul respectiv, s-a folosit metoda "Substring".

 **Comanda iesiri digitale**   
  
    Urmatoarea aplicatie utilizeaza iesirile digitale:

|  |
| --- |
| namespace RS\_232\_v00  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  string txt, cda;  int k;  static String[] Nume\_porturi = new String[11];  private void Form1\_Load(object sender, EventArgs e)  {  int i, j;  // Listez porturile seriale  Nume\_porturi = System.IO.Ports.SerialPort.GetPortNames();  this.Porturi\_s.Items.Clear();  //Adaug porturile exixtente  for (i = 0; i < Nume\_porturi.Length; i++)  {  this.Porturi\_s.Items.Add(Nume\_porturi[i]);  }  //Pozitionarea listei pe primul element  this.Porturi\_s.SelectedIndex = 0;  this.label1.Text = "Conectati-va la portul serial!";  }  private void but\_con\_Click(object sender, EventArgs e)  {  if (!this.serialPort1.IsOpen)  {  this.serialPort1.PortName = System.Convert.ToString(this.Porturi\_s.Items[this.Porturi\_s.SelectedIndex]);  this.serialPort1.Open();  this.label1.Text = "Portul a fost deschis";  }  else  {  this.label1.Text = "Portul este deja deschis deschis";  }  this.serialPort1.Write("O77");  }  private void but\_dec\_Click(object sender, EventArgs e)  {  this.serialPort1.Close();  this.label1.Text = "Portul este deconectat!";  }  private void timer1\_Tick(object sender, EventArgs e)  {  k++;  if (k > 123456789)  k = 0;  this.label2.Text = k.ToString();  cda = "O" + System.Convert.ToString(this.numericUpDown1.Value);  if (this.serialPort1.IsOpen)  {  this.serialPort1.Write(cda);  txt = "";  txt = this.serialPort1.ReadExisting();  if (txt.Length > 0)  {  this.textBox2.Text = txt;  }  else  {  this.label1.Text = "Nu vin date!";  }  }  }  }  } |