```
/*
    Programma : cppContatti

    Autore : Riontino Raffaele 4 informatici (corso serale)
             ITIS Molinari Milano   17/05/2011

    Funzionalità : gestione di una rubrica telefonica con esportazione dei dati su
                   file XML o CSV. Permette di importare i dati da file con estensione CSV.

    Compilatore : Misrosoft Visual C++ 2010 Express
*/

#include "stdafx.h"
#include <iostream>
#include <vector>
#include <string>
#include <conio.h>
#include <exception>
using namespace std;


class Contatto
      {
       public : Contatto () {this -> nome = ""; this -> cognome = "";this -> telefono = "";this -
> email = "";};
                Contatto (string nome = "", string cognome = "", string telefono = "" , string
email = "") {this->nome = nome; this->cognome = cognome; this->telefono = telefono; this->email
= email;};
                void setDati (string nome = "", string cognome = "", string telefono = "" ,
string email = "") {this->nome = nome; this->cognome = cognome; this->telefono = telefono; this
>email = email;};
                void setNome (string nome) {this -> nome = nome;};
                void setCognome (string cognome) {this -> cognome = cognome;};
                void setTelefono (string telefono) {this -> telefono = telefono;};
                void setEmail (string email) {this -> email = email;};
                string getNome() {return this -> nome;};
                string getCognome () {return this -> cognome;};
                string getTelefono () {return this -> telefono;};
                string getEmail () {return this -> email;};
                string toString () {return  "\n\tCognome : " + this -> cognome + "\n\tNome : " +
this -> nome + "\n\tTelefono : " + this -> telefono + "\n\tE-Mail : " + this -> email;};
       private : string nome , cognome , telefono , email;
      }; //fine classe contatto


typedef vector<Contatto> VetC;

void titolo ();

char menu (int);

void aggiungiContatto (VetC &);

void visualizzaElenco (VetC);

void salvaFile (string , VetC);

bool leggiFile (string , VetC &);

void caricaContatto (string, VetC &);

bool controllaEsistente(string);

void creaApriFile(string & ,bool & ,bool & ,VetC &);

bool controllaCaratteri (string);

void cercaContatto (VetC);

bool modificaContatto (VetC & );
```

```
bool eliminaContatto (VetC &);

void esportaCSV (string , VetC);

void esportaXML (string , VetC);


int main ()
    {
     VetC elenco;
     char scelta;
     bool aperto = false , modificato = false;
     string nomeFile ="";
     do{
         scelta = menu (elenco.size());
         switch (scelta)
                {
                 case 'n' : creaApriFile(nomeFile , aperto , modificato , elenco);
                            break;
                 case 'a' : if (!aperto)
                               {
                                cout << "\n\n\t\aPrima di aggiungere contatti devi creare un
nuovo file\n\to aprire un file creato in precedenza!!";
                                getch();
                               }
                            else
                                 {
                                  titolo ();
                                  cout << "\n\n\tNuovo contatto :\n\n";
                                  aggiungiContatto (elenco);
                                  modificato = true;
                                 }
                            break;
                 case 'v' : if (!aperto)
                               {
                                cout << "\n\n\t\aNessun file aperto!!";
                                getch();
                               }
                            else visualizzaElenco(elenco);
                            break;
                 case 's' : if (aperto)
                               {
                                salvaFile(nomeFile , elenco);
                                modificato = false;
                               }
                            else
                                 {
                                  cout << "\n\n\t\aDevi prima aprire o creare un file !!";
                                  getch();
                                 }
                            break;
                 case 'm' : if (!aperto)
                               {
                                cout << "\n\n\t\aNessun file aperto!!";
                                getch();
                               }
                            else
                                 modificato =  modificaContatto(elenco);
                            break;//modifica
                 case 'c' : if (!aperto)
                               {
                                cout << "\n\n\t\aNessun file aperto!!";
                                getch();
                               }
                            else
                                cercaContatto (elenco);
                            break;//cerca
                 case 'd' : if (!aperto)
                               {
```

```
                                cout << "\n\n\t\aNessun file aperto!!";
                                getch();
                               }
                            else
                                 modificato =  eliminaContatto(elenco);
                            break;//elimina
                 case 'w' : if (!aperto)
                               {
                                cout << "\n\n\t\aNessun file aperto!!";
                                getch();
                               }
                            else esportaCSV (nomeFile , elenco);
                            break;
                 case 'x' : if (!aperto)
                               {
                                cout << "\n\n\t\aNessun file aperto!!";
                                getch();
                               }
                            else esportaXML (nomeFile , elenco);
                            break;
                }
        }while (scelta != 'e');
     if (modificato)
        {
         titolo ();
         cout << "\n\n\t\aGli ultimi dati non sono stati salvati.Vuoi salvarli (s / n)?";
         do
           scelta = getch();
           while (scelta != 's' && scelta != 'n');
         if (scelta == 's') salvaFile(nomeFile , elenco);
        }
     return 0;
    }//fine main



void titolo()
     {
      system("cls");
      cout << "\n\t\t\tGestione contatti\n";
     }//fine main


char menu(int dim)
     {
      titolo ();
      cout << "\n\tcontatti presenti nell'elenco : " << dim << "\n\n\t\tMenu'\n\n";
      cout << "\n\tn - apri / crea File";
      cout << "\n\ta - aggiungi contatto";
      cout << "\n\tc - cerca contatto";
      cout << "\n\tm - modifica contatto";
      cout << "\n\td - elimina contatto";
      cout << "\n\tv - visualizza elenco";
      cout << "\n\ts - salva dati su file";
      cout << "\n\tx - esporta XML";
      cout << "\n\tw - esporta CSV";
      cout << "\n\te - esci\n";
      cout << "\n\tscelta : ";
      return getch();
     }//fine menu


void aggiungiContatto (VetC & elenco)
     {
      //titolo ();
      string temp;
      Contatto *nuovo = new Contatto("" , "" , "" , "");
      //cout << "\n\n\tNuovo contatto :\n\n";
      cout << "\n\tNome : ";
      do
```

```
        getline(cin , temp);
        while(!controllaCaratteri(temp));
      nuovo->setNome(temp);
      cout << "\tCognome : ";
      do
        getline(cin , temp);
        while(!controllaCaratteri(temp));
      nuovo->setCognome(temp);
      cout << "\tTelefono : ";
      do
        getline(cin , temp);
        while(!controllaCaratteri(temp));
      nuovo->setTelefono(temp);
      cout << "\tE-Mail : ";
      do
        getline(cin , temp);
        while(!controllaCaratteri(temp));
      nuovo->setEmail(temp);
      int pos=-1; //conterrà la posizione in cui inserire il dato
      for(int i = 0; pos < 0 && i < elenco.size() ; i++) //scorro il vector
              if (elenco.at(i).getCognome() > nuovo->getCognome() || elenco.at(i).getCognome()
== nuovo->getCognome() && elenco.at(i).getNome() >= nuovo->getNome())
                 pos = i;      //salvo la posizione
      if(pos >= 0) //se è stata trovata la posizione
             elenco.insert(elenco.begin()+pos,*nuovo);  //inserisco il dato con il metodo insert
      else
          elenco.push_back(*nuovo); //altrimenti in coda
     }//fine aggiungiContatto


void visualizzaElenco (VetC elenco)
     {
      titolo ();
      for (int i = 0 ; i < elenco.size() ; i++)
          {
           cout << "\n\n\tcontatto n." << i << endl << elenco.at(i).toString();
          }
      cout << "\n\n\ttotale contatti : " << elenco.size();
      getch();
     }//fine visualizzaElenco


void salvaFile(string nomeFile , VetC elenco)
     {
      //salvataggio dei dati in formato csv
      FILE *file;
      if (file = fopen(nomeFile.c_str() , "wt"))
         {
          for (int i = 0 ; i < elenco.size() ; i++)
              {
               fprintf(file , "%s;%s;%s;%s\n" , elenco.at(i).getNome().c_str(),
elenco.at(i).getCognome().c_str() , elenco.at(i).getTelefono().c_str() ,
elenco.at(i).getEmail().c_str());
              }
          fclose(file);
          cout << "\n\n\tSalvati " << elenco.size() << " contatti nel file '" << nomeFile << "'
!!";
         }
      else cout << "\n\t\aErrore durante la scrittura dei dati su file!!!";
      getch();
     }//fine salvaFile


bool leggiFile (string nomeFile , VetC & elenco)
     {
      FILE * file;
      if (file = fopen(nomeFile.c_str() , "rt"))
         {
          try {
              char ch;
```

```
              string riga = "";
              while ((ch = fgetc (file)) != EOF)
                {
                 if (ch != '\n')
                    riga += ch;
                 else
                    {
                     riga += ch;
                     caricaContatto(riga , elenco); //se no riesce a separare i dati solleva
un'eccezione
                     riga = "";
                    }
                } //fine while
               fclose(file);
               return true;
              } //fine try
          catch (string e)
                {
                 cerr << "\n\t\a" << e;
                 elenco.clear();
                 return false;
                } //fine catch
         }//fine if
      else
           {
            cout << "\n\n\tErrore durante l'apertura del file!!";
            getch();
            return false;
            }
     }//fine leggiFile


void caricaContatto (string riga, VetC & elenco)
     {
               //struttura della riga  "nome;cognome;telefono;email\n";
               string err = "file non compatibile!!"; //messaggio dell'eccezione
               Contatto *nuovo = new Contatto ("","","","");
               string temp ;
               if (riga.find(";") > riga.size() || riga.find(";") < 0) //se non trova il ;
solleva un'eccezione
                  throw (err);
               temp = riga.substr(0 , riga.find(";")); //ricavo il nome
               nuovo->setNome(temp);
               riga = riga.substr(riga.find(";") + 1); //elimino il nome dalla stringa
               if (riga.find(";") > riga.size() || riga.find(";") < 0) //se non trova il ;
solleva un'eccezione
                  throw (err);
               temp = riga.substr(0 , riga.find(";")); //ricavo il cognome
               nuovo->setCognome(temp);
               riga = riga.substr(riga.find(";") + 1); //elimino il cognome dalla stringa
               if (riga.find(";") > riga.size() || riga.find(";") < 0) //se non trova il ;
solleva un'eccezione
                  throw (err);
               temp = riga.substr(0 , riga.find(";")); //ricavo il telefono
               nuovo->setTelefono(temp);
               riga = riga.substr(riga.find(";") + 1); //elimino il telefono dalla stringa
               if (riga.find("\n") > riga.size() || riga.find("\n") < 0) //se non trova il \n
solleva un'eccezione
                  throw (err);
               temp = riga.substr(0 , riga.find("\n")); //ricavo la email
               nuovo->setEmail(temp);
               cout << nuovo->toString();
               int pos=-1; //conterrà la posizione in cui inserire il dato
               for(int i = 0; pos < 0 && i < elenco.size() ; i++) //scorro il vector
                       if (elenco.at(i).getCognome() > nuovo->getCognome() ||
elenco.at(i).getCognome() == nuovo->getCognome() && elenco.at(i).getNome() >= nuovo->getNome())
                               pos = i;      //salvo la posizione
               if(pos >= 0) //se è stata trovata la posizione
                 elenco.insert(elenco.begin()+pos,*nuovo);  //inserisco il dato con il metodo
insert
```

```
                else
                elenco.push_back(*nuovo);    //altrimenti in coda
     }//fine caricaContatto


bool controllaEsistente (string temp) //controlla se esiste un file
     {
      FILE * f;
      if (f = fopen(temp.c_str() , "rt"))
         {
          fclose(f);
          return true;
         }
      else return false;
     }//fine controllaEsistente


void creaApriFile(string & nomeFile ,bool & aperto ,bool & modificato ,VetC & elenco)
     {
      titolo();
      char scelta;
      FILE * nuovo;
      if (aperto && modificato)
         {
         cout << "\n\t\aIl file su cui si stava lavorando verra' chiuso.";
         cout << "\n\tSalvare le ultime modifiche (s / n)?";
         do
           scelta = getch();
           while (scelta != 's' && scelta != 'n');
         if (scelta == 's')
            salvaFile(nomeFile , elenco);
         }
      modificato = false;
      aperto = false;
      elenco.clear(); //svuoto il vector
      titolo();
      cout << "\n\n\t\tNome del file da aprire o da creare : ";
      getline(cin , nomeFile);
      if (controllaEsistente(nomeFile)) //se il file esiste già
         {
          if (leggiFile(nomeFile , elenco)) //e lo ricarico con i dati presenti sul file da
aprire
             {
             cout << "\n\n\tCaricati " << elenco.size() << " contatti dal file " << nomeFile <<
" !";
             aperto = true;
             }
          getch();
          }
         else {
               char scelta;
               cout << "\n\n\tCreare il file '" << nomeFile << "' file (s / n) ?";
               do
                scelta = getch ();
                while (scelta != 's' && scelta != 'n');
               if (scelta == 's')
                   {
                    if (nuovo = fopen(nomeFile.c_str() , "wt"))
                      {
                       cout << "\n\n\tFile creato " << nomeFile << endl;
                       fclose(nuovo);
                       aperto = true;
                       }
                    else
                        cout << "\n\n\t\aErrore durante la creazione del file!!";
                   }
               else cout << "\n\t\aNessun file aperto o creato!!";
                getch();
         }
     }//fine creaApriFile
```

```
bool controllaCaratteri (string temp)
    {
        if (temp.find(";") > -1 || temp.find(";") < (temp.size() +1) || temp.size() < 1)
        {
         cout << "\t\t\aValore non corretto...Ripeti!!\n";
         return false;
        }
     else return true;
    }//fine controllaCaratteri


void cercaContatto (VetC elenco)
    {
     string temp;
     titolo ();
     cout << "\n\n\tNome , cognome , telefono , o email del contatto da cercare :\n\t";
     do
       getline(cin , temp);
       while (!controllaCaratteri(temp));
    int conto = 0;
    for (int i = 0 ; i < elenco.size() ; i++)
        {
            if (elenco.at(i).getCognome().find(temp) > -1 ||
elenco.at(i).getCognome().find(temp) < elenco.size()+1
                || elenco.at(i).getNome().find(temp) > -1 || elenco.at(i).getNome().find(temp) <
elenco.size()+1
                || elenco.at(i).getTelefono().find(temp) > -1 ||
elenco.at(i).getTelefono().find(temp) < elenco.size()+1
                || elenco.at(i).getEmail().find(temp) > -1 || elenco.at(i).getEmail().find(temp)
< elenco.size()+1 )
                {
                 cout << "\n\tn." << i << elenco.at(i).toString() << endl;
                 conto++;
                }
        }
    cout << "\n\tContatti trovati : " << conto;
    getch();
    }//fine cercaVontatto


bool modificaContatto (VetC & elenco)
    {
     string temp;
     char scelta;
     titolo ();
     cout << "\n\n\tNome , cognome , telefono o email del contatto da modificare :\n\t";
     do
       getline(cin , temp);
       while (!controllaCaratteri(temp));
    unsigned int i = 0;
    bool flag = true , trovato = false;
    while (i < elenco.size() && flag)
        {
            if (elenco.at(i).getCognome().find(temp) > -1 ||
elenco.at(i).getCognome().find(temp) < elenco.size()+1
                || elenco.at(i).getNome().find(temp) > -1 || elenco.at(i).getNome().find(temp) <
elenco.size()+1
                || elenco.at(i).getTelefono().find(temp) > -1 ||
elenco.at(i).getTelefono().find(temp) < elenco.size()+1
                || elenco.at(i).getEmail().find(temp) > -1 || elenco.at(i).getEmail().find(temp)
< elenco.size()+1 )
                {
                 cout << "\n\tn." << i << elenco.at(i).toString() << endl;
                 cout << "\n\tModificare questo contatto (s / n) ? ";
                 do
                    scelta = getch();
                 while (scelta != 's' && scelta != 'n');
                 trovato = true;
```

```
                if (scelta == 's')
                   {
                    flag = false;
                    cout << endl;
                    elenco.erase(elenco.begin()+i);
                    aggiungiContatto (elenco);
                    cout << "\n\tContatto modificato !!";
                    getch();
                   }
                }
         i++;
        }
     if (!flag) return true;
     else
         {
          if (!trovato) cout << "\n\n\tNessun contatto trovato !!";
          else cout << "\n\n\tNessun contatto Modificato !!";
          getch();
          return false;
         }
    }//fine modificaContatto


bool eliminaContatto (VetC & elenco)
    {
     string temp;
     char scelta;
     titolo ();
     cout << "\n\n\tNome , cognome , telefono o email del contatto da eliminare :\n\t";
     do
       getline(cin , temp);
       while (!controllaCaratteri(temp));
    unsigned int i = 0;
    bool flag = true , trovato = false;
    while (i < elenco.size() && flag)
        {
            if (elenco.at(i).getCognome().find(temp) > -1 ||
elenco.at(i).getCognome().find(temp) < elenco.size()+1
                || elenco.at(i).getNome().find(temp) > -1 || elenco.at(i).getNome().find(temp) <
elenco.size()+1
                || elenco.at(i).getTelefono().find(temp) > -1 ||
elenco.at(i).getTelefono().find(temp) < elenco.size()+1
                || elenco.at(i).getEmail().find(temp) > -1 || elenco.at(i).getEmail().find(temp)
< elenco.size()+1 )
                {
                 cout << "\n\tn." << i << elenco.at(i).toString() << endl;
                 cout << "\n\tEliminare questo contatto (s / n) ? ";
                 trovato = true;
                 do
                    scelta = getch();
                 while (scelta != 's' && scelta != 'n');
                 if (scelta == 's')
                    {
                     flag = false;
                     cout << endl;
                     elenco.erase(elenco.begin()+i);
                     cout << "\n\tContatto eliminato !!";
                     getch();
                    }
                }
         i++;
        }
     if (!flag) return true;
     else
         {
           if (!trovato)
               cout << "\n\n\tNessun contatto tovato !!";
           else cout << "\n\n\tNessun contatto eliminato !!";
           getch();
           return false;
```

```
        }
    }//fine eliminaContatto


void esportaCSV(string nomeFile , VetC elenco)
    {
     FILE * file ;
     if (nomeFile.find_last_of(".") > -1 || nomeFile.find_last_of(".") < nomeFile.size()+1)
         nomeFile = nomeFile.substr(0 , nomeFile.find_last_of("."));
     nomeFile = nomeFile + ".csv";
     if (file = fopen(nomeFile.c_str() , "wt"))
        {
            for (int i = 0 ; i < elenco.size() ; i++)
                fprintf(file , "%s;%s;%s;%s\n" , elenco.at(i).getNome().c_str() ,
elenco.at(i).getCognome().c_str() , elenco.at(i).getTelefono().c_str() ,
elenco.at(i).getEmail().c_str());
            fclose (file);
            cout << "\n\n\tCreato il file '" << nomeFile << "' !!";
            system(nomeFile.c_str());
        }
     else cout << "\n\terrore durante la creazione del file CSV";
     getch ();
    }//fine esportaCSV


void esportaXML (string nomeFile , VetC elenco)
    {
     FILE * file ;
     if (nomeFile.find_last_of(".") > -1 || nomeFile.find_last_of(".") < nomeFile.size()+1)
         nomeFile = nomeFile.substr(0 , nomeFile.find_last_of("."));
     nomeFile = nomeFile + ".xml";
     if (file = fopen(nomeFile.c_str() , "wt"))
        {
            fprintf(file , "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<RUBRICA>");
            for (int i = 0 ; i < elenco.size() ; i++)
                fprintf(file ,
"<CONTATTO><NOME>%s</NOME><COGNOME>%s</COGNOME><TELEFONO>%s</TELEFONO><EMAIL>%s</EMAIL></CONTATTO
, elenco.at(i).getNome().c_str() , elenco.at(i).getCognome().c_str() ,
elenco.at(i).getTelefono().c_str() , elenco.at(i).getEmail().c_str());
            fprintf(file , "</RUBRICA>");
            fclose (file);
            cout << "\n\n\tCreato il file '" << nomeFile << "' !!";
            system(nomeFile.c_str());
        }
     else cout << "\n\terrore durante la creazione del file XML";
     getch ();
    }//fine esportaXML
```