# Code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace AddressBinary

{

static class AddressBinary

{

struct ContactDetails

{

public string firstname;

public string surname;

public string email;

public string phone;

public string postcode;

public string salutation;

}

static int Menu()

{

Console.WriteLine();

Console.WriteLine("Please choose an option: ");

Console.WriteLine("Add contact: 1");

Console.WriteLine("Display contact: 2");

Console.WriteLine("Quit: 9");

Console.WriteLine();

string Option = (Console.ReadLine());

return Convert.ToInt32(Option);

}

static ContactDetails EnterContact() // returns a struct containing the data for the new contact

{

ContactDetails newContact = new ContactDetails(); //create the struct

Console.WriteLine("Enter new contact details:");

Console.Write("First Name: ");

newContact.firstname = Console.ReadLine();

Console.Write("Surname: ");

newContact.surname = Console.ReadLine();

Console.Write("email: ");

string emailInput = Console.ReadLine();

while (!(ValidateEmail(emailInput)))

{

Console.WriteLine("Invalid email format: please try again");

emailInput = Console.ReadLine();

}

newContact.email = emailInput;

Console.Write("Phone: ");

string phoneInput = Console.ReadLine();

while (!(ValidatePhone(emailInput)))

{

Console.WriteLine("Invalid phone format: please try again");

emailInput = Console.ReadLine();

}

newContact.phone = phoneInput;

Console.Write("Postcode: ");

newContact.postcode = Console.ReadLine();

Console.Write("Salutation: ");

newContact.salutation = Console.ReadLine();

return newContact;

}

static bool ValidatePhone(string phone)

{

//remove spaces and check

phone = RemoveWhitespace(phone);

return phone.All(Char.IsDigit);

}

public static string RemoveWhitespace(this string input)

{

return new string(input.ToCharArray()

.Where(c => !Char.IsWhiteSpace(c))

.ToArray());

}

static bool ValidateEmail(string email)

{

int atCount = 0;

for (int i = 0; i < email.Length; i++) //does email contain just one @?

{

if (email[i] == '@')

{

atCount++;

}

}

if (atCount == 1)

{

int dotCount = 0;

for (int i = 0; i < email.Length; i++) //does email contain and . ?

{

if (email[i] == '.')

{

dotCount++;

}

}

if (dotCount >= 1)

{

return true;

}

else

{

return false;

}

}

else

{

return false;

}

}

static void DisplayContact(ContactDetails contact)

{

Console.WriteLine("Name: " + contact.salutation + " " + contact.firstname + " " + contact.surname);

Console.WriteLine("email: " + contact.email);

Console.WriteLine("Phone: " + contact.phone);

Console.WriteLine("PostCode: " + contact.postcode);

}

static void WriteBinaryFile(ContactDetails contact)

{

Console.Write("Enter the full file path of the location you would like the file to be saved: ");

string filePath = Console.ReadLine();

using (BinaryWriter writer = new BinaryWriter(File.Open(@filePath, FileMode.Create)))

{

writer.Write(contact.firstname);

writer.Write(contact.surname);

writer.Write(contact.email);

writer.Write(contact.phone);

writer.Write(contact.postcode);

writer.Write(contact.salutation);

}

}

static ContactDetails ReadBinaryFile()

{

Console.Write("Enter the full file path of the file to open: ");

string filePath = Console.ReadLine();

using (BinaryReader reader = new BinaryReader(File.Open(@filePath, FileMode.Open)))

{

//first two strings in file are first name and surname

ContactDetails contact;

contact.firstname = reader.ReadString();

contact.surname = reader.ReadString();

contact.email = reader.ReadString();

contact.phone = reader.ReadString();

contact.postcode = reader.ReadString();

contact.salutation = reader.ReadString();

return contact;

}

}

public static void Main(string[] args)

{

int choice = 0;

do

{

choice = Menu();

switch (choice)

{

case 1:

// create a local Contact Details structure and fill it with details

ContactDetails newContact;

newContact = EnterContact();

WriteBinaryFile(newContact); // save it to a binary file

break;

case 2:

ContactDetails contact = ReadBinaryFile();

DisplayContact(contact);

break;

}

}

while ((choice != 9));

}

}

}

# Code Running

