**Course Project   
DeVry University  
College of Engineering and Information Sciences**

***Screenshot of program running:***

A screenshot of a computer

Description automatically generated

***Form code (only the code for the form and classes, not program.cs):***

MainForm.cs using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

namespace Rodriguez\_CourseProject\_Part2

{

public partial class MainForm: Form

{

// class level references

private const string FILENAME = "Employee.dat";

private object updateform;

private int lifeInsurance;

private object updateForm;

private DateTime hireDate;

private int vacation;

private Salary sal;

public MainForm()

{

InitializeComponent();

}

private void AddButton\_Click(object sender, EventArgs e)

{

// add item to the Employee listbox

InputForm frmInput = new InputForm();

using (frmInput)

{

DialogResult result = frmInput.ShowDialog();

// see if the input form was cancelled

if (result == DialogResult.Cancel)

return; // end the method since cancelled

// get user's input and create Employee object

string fName = frmInput.FirstNameTextBox.Text;

string lName = frmInput.LastNameTextBox.Text;

string ssn = frmInput.SSNTextBox.Text;

string date = frmInput.HireDateTextBox.Text;

DateTime hireDate = DateTime.Parse(date);

string healthIns = frmInput.HealthInsuranceTextBox.Text;

int lifeIns = int.Parse(frmInput.LifeInsuranceTextBox.Text);

int vacation = int.Parse(frmInput.VacationDaysTextBox.Text);

Benefits ben = new Benefits(healthIns, lifeIns, vacation);

Employee emp = null; // empty reference

if( frmInput.SalaryRadioButton.Checked)

{

double salary = double.Parse(frmInput.SalaryTextBox.Text);

emp = new Salary(fName, lName, ssn, hireDate, ben, salary);

}

else if(frmInput.HourlyRadioButton.Checked )

{

double hourlyRate = double.Parse(frmInput.HourlyRateTextBox.Text);

double hoursWorked = double.Parse(frmInput.HoursWorkedTextBox.Text);

emp = new Hourly(fName, lName, ssn, hireDate, ben, hourlyRate, hoursWorked);

}

else

{

MessageBox.Show("Error. Please select an employee type.");

return; // end the method

}

// add the Employee object to the Employee listbox

EmployeesListBox.Items.Add(emp);

// write all date to the file

WriteEmpsToFile();

}

}

private void WriteEmpsToFile()

{

// covert the listbox items to a generic list

List<Employee> empList = new List<Employee>();

foreach(Employee emp in EmployeesListBox.Items)

{

empList.Add(emp);

}

// open a pipe to the file and create a translator

FileStream fs = new FileStream(FILENAME, FileMode.Create);

BinaryFormatter formatter = new BinaryFormatter();

// write the generic list to the file

formatter.Serialize(fs, empList);

// close the pipe

fs.Close();

}

private void RemoveButton\_Click(object sender, EventArgs e)

{

// remove the selected item from the employee listbox

int itemNumber = EmployeesListBox.SelectedIndex;

if (itemNumber > -1)

{

EmployeesListBox.Items.RemoveAt(itemNumber);

}

else

{

MessageBox.Show("Please select employee to remove.");

}

WriteEmpsToFile();

}

private void DisplayButton\_Click(object sender, EventArgs e)

{

ReadEmpsFromFile();

}

private void ReadEmpsFromFile()

{

// check to see if file exists

if (File.Exists(FILENAME) && new FileInfo(FILENAME).Length > 0)

{

// create a pipe from the file and create the "translator"

FileStream fs = new FileStream(FILENAME, FileMode.Open);

BinaryFormatter formatter = new BinaryFormatter();

// read all Employee objects from the file

List<Employee> list = (List<Employee>)formatter.Deserialize(fs);

// close the pipe

fs.Close();

// clear ListBox items and copy the file's Employee objects into our ListBox

EmployeesListBox.Items.Clear();

foreach (Employee emp in list)

EmployeesListBox.Items.Add(emp);

}

}

private void PrintPaychecksButton\_Click(object sender, EventArgs e)

{

foreach (Employee emp in EmployeesListBox.Items)

{

string line1 = "pay to: " + emp.FirstName + " " + emp.LastName;

string line2 = "Amount Of: " + emp.CalculatePay().ToString("C2");

string output = "Paycheck:\n\n" +line1 + "\n" + line2;

MessageBox.Show(output);

}

}

private void EmployeesListBox\_DoubleClick(object sender, EventArgs e)

{

// get the selected Employee object

Employee emp = EmployeesListBox.SelectedItem as Employee;

if (emp != null)

{

}

// show the Input/Update form with the Employee info

InputForm Updateform = new InputForm();

Updateform.SubmitButton.Text = "Update";

Updateform.StartPosition = FormStartPosition.CenterParent;

Updateform.FirstNameTextBox.Text = emp.FirstName;

Updateform.LastNameTextBox.Text = emp.LastName;

Updateform.SSNTextBox.Text = emp.SSN;

Updateform.HireDateTextBox.Text = emp.HireDate.ToString();

Updateform.HealthInsuranceTextBox.Text = emp.BenefitsEmp.HealthInsuraance;

Updateform.LifeInsuranceTextBox.Text = emp.BenefitsEmp.LifeInsurance.ToString();

Updateform.VacationDaysTextBox.Text = emp.BenefitsEmp.Vacation.ToString();

// check to see if emp is a Salary or Hourly object

if(emp is Salary)

{

Updateform.HourlyRateLabel.Visible = false;

Updateform.HourlyRateTextBox.Visible = false;

Updateform.HoursWorkedLabel.Visible = false;

Updateform.HoursWorkedTextBox.Visible = false;

Updateform.SalaryLabel.Visible = true; // show this one

Updateform.SalaryTextBox.Visible = true; // show this one

// mark the Salary radio button as checked

Updateform.SalaryRadioButton.Checked = true;

// convert the Employee object to salary object

Salary sal = (Salary)emp;

// show the Salary attribute

Updateform.SalaryTextBox.Text = sal.AnnualSalary.ToString("F0");

}

else if(emp is Hourly )

{

Updateform.HourlyRateLabel.Visible = false; // hide this one

Updateform.HourlyRateTextBox.Visible = false; // hide this one

Updateform.HoursWorkedLabel.Visible = true;

Updateform.HoursWorkedTextBox.Visible = true;

Updateform.SalaryLabel.Visible = true;

Updateform.SalaryTextBox.Visible = true;

// mark the Hourly radiobutton as checked

Updateform.HourlyRadioButton.Checked = true;

// convert the Employee reference to a Hourly object

Hourly hrly = (Hourly)emp;

// show the Hourly information

Updateform.HourlyRateTextBox.Text = hrly.HourlyRate.ToString("F2");

Updateform.HoursWorkedTextBox.Text = hrly.HoursWorked.ToString("F2");

}

else

{

MessageBox.Show("Error. Invalid employee type found.");

return; // end the method

}

DialogResult result = Updateform.ShowDialog();

// if can cancelled, stop the method

if (result == DialogResult.Cancel)

return; // end the method

// delete the selected object

int position = EmployeesListBox.SelectedIndex;

EmployeesListBox.Items.RemoveAt(position);

// create new employee using the updated information

Employee newEmp = null;

string fName = Updateform.FirstNameTextBox.Text;

string lName = Updateform.LastNameTextBox.Text;

string ssn = Updateform.SSNTextBox.Text;

DateTime HireDate = DateTime.Parse(Updateform.HireDateTextBox.Text);

string healthInsurance = Updateform.HealthInsuranceTextBox.Text;

int LifeInsurance = int.Parse(Updateform.LifeInsuranceTextBox.Text);

int Vacation = int.Parse(Updateform.VacationDaysTextBox.Text);

Benefits ben = new Benefits(healthInsurance, lifeInsurance, vacation);

if (Updateform.SalaryRadioButton.Checked)

{

double salary = double.Parse(Updateform.SalaryTextBox.Text);

newEmp = new Salary(fName, lName, ssn, hireDate, ben, salary);

}

else if (Updateform.HourlyRadioButton.Checked )

{

double hourlyRate = double.Parse(Updateform.HourlyRateTextBox.Text);

double hoursWorked = double.Parse(Updateform.HoursWorkedTextBox.Text);

newEmp = new Hourly(fName,lName,ssn,hourlyRate,hoursWorked);

}

else

{

MessageBox.Show("Error.Invalid employee type");

return; // end the method

}

// add the new employee to the list box

EmployeesListBox.Items.Add(newEmp);

}

}

}

Salary.cs using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Rodriguez\_CourseProject\_Part2

{

[Serializable]

public class Salary : Employee

{

// attributes

private double annualSalary;

// constructors

public Salary() : base()

{

annualSalary = 0.00;

}

public Salary(string firstname, string lastname,

string ssn, DateTime hireDate, Benefits benefits,

double annualSalary)

: base(firstname,lastname, ssn, hireDate,benefits)

{

this.annualSalary = annualSalary;

}

// behaviors

public override double CalculatePay()

{

return annualSalary / 26.0;

}

public override string ToString()

{

return base.ToString() + ", annualSalary="

+ annualSalary.ToString("C2");

}

// properties

public double AnnualSalary

{

get { return annualSalary; }

set { annualSalary = value; }

}

}

}

Benefits.csusing System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Rodriguez\_CourseProject\_Part2

{

[Serializable]

public class Benefits

{

// attributes

private string healthInsuraance;

private int lifeInsurance;

private int vacation;

// constructors

public Benefits()

{

healthInsuraance ="N/A";

lifeInsurance = 0;

vacation = 0;

}

public Benefits(string healthInsuraance, int lifeInsurance, int vacation)

{

this.healthInsuraance = healthInsuraance;

this.lifeInsurance = lifeInsurance;

this.vacation = vacation;

}

// behaviors

public override string ToString()

{

return "healthInsurance=" + healthInsuraance

+ ", lifeInsurance=" + lifeInsurance

+ ", vacation=" + vacation;

}

// properties

public string HealthInsuraance

{

get { return healthInsuraance; }

set { healthInsuraance = value; }

}

public int LifeInsurance

{

get { return lifeInsurance; }

set { lifeInsurance = value;}

}

public int Vacation

{

get{ return vacation; }

set { vacation = value; }

}

public string HealthInsurance { get; internal set; }

}

}

Employee.cs using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.CompilerServices;

using System.Runtime.ExceptionServices;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Threading.Tasks;

namespace Rodriguez\_CourseProject\_Part2

{

[Serializable]

public abstract class Employee

{

// attributes

private string firstName;

private string lastName;

private string ssn;

private DateTime hireDate;

private Benefits benefits;

// constructors

public Employee()

{

firstName = "N/A";

lastName = "N/A";

ssn = "N/A";

hireDate = new DateTime();

benefits = new Benefits();

}

public Employee(string firstname, string lastname,

string ssn, DateTime hireDate, Benefits benefits)

{

this.firstName = firstname;

this.lastName = lastname;

this.ssn = ssn;

this.hireDate = hireDate;

this.benefits = benefits;

}

// behaviors

public override string ToString()

{

return "firstname=" + firstName

+ ", lastName=" + lastName

+ ", ssn=" + ssn

+ ", hireDate=" + hireDate.ToShortDateString()

+ ", healthInsurance=" + benefits.HealthInsuraance

+ ", lifeInsurance=" + benefits.LifeInsurance

+ ", vacation=" + benefits.Vacation;

}

public abstract double CalculatePay();

// properties

public string FirstName

{

get { return firstName; }

set { firstName = value; } // emp.FirstName = "Bob";

}

public string LastName

{

get { return lastName; }

set { lastName = value; }

}

public string SSN

{

get { return ssn; }

set { ssn = value; }

}

public DateTime HireDate

{

get { return hireDate; }

set { hireDate = value; }

}

public Benefits BenefitsEmp

{

get { return benefits; }

set { benefits = value; }

}

}

}