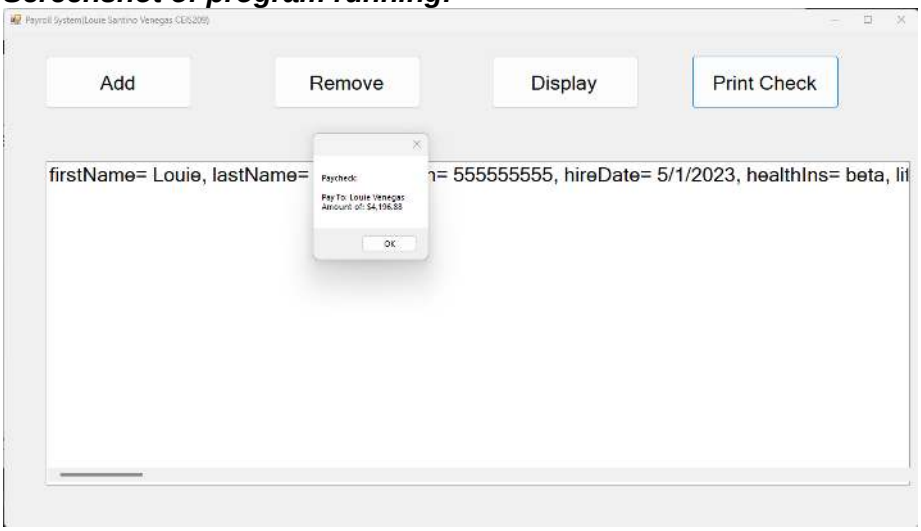
Course Project
DeVry University
College of Engineering and Information Sciences

Screenshot of program running:



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.Drawing.Text;
using System.IO;
using System.Linq;
using System.Runtime.Serialization.Formatters.Binary;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Venegas_CourseProject_Part2
{
    public partial class MainForm : Form
```

```
{
    //form level refrences
    private const string FILENAME = "Employees.dat";

    public MainForm()
    {
        InitializeComponent();
    }

    private void AddButton_Click(object sender, EventArgs e)
    {
        //add item tp the employee list box
        EmployeeInputForm frmInput = new EmployeeInputForm();

        using (frmInput)
        {
            DialogResult result = frmInput.ShowDialog();

            //see if input for was exited with exit button
            if (result == DialogResult.Cancel)
                return;  // <== ends method since canceled\

            //get the employee object from the input form
            string fName = frmInput.FirstNameTextBox.Text;
            string lName = frmInput.LastNameTextBox.Text;
            string ssn = frmInput.SSNTextBox.Text;
            string date = frmInput.DateOfHireTextBox.Text;
            DateTime hireDate =DateTime.Parse(date);
            string healthInsurance = frmInput.HealthInsTextBox.Text;
            int lifeInsurance = int.Parse(frmInput.LifeInsTextBox.Text);
            int vacation = int.Parse(frmInput.VacationTextBox.Text);

            //create composite object
            Benefits ben = new Benefits(healthInsurance, lifeInsurance, vacation);

            //create a new employee reference
            Employee emp = null;

            //create child object based on selected radio button
            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("Please select a employee type");
            return;
        }

        //add the employee to the list box

        EmployeeListBox.Items.Add(emp);

        //write the employee to the file
        WriteEmpsToFile();

    }



}



private void RemoveButton_Click(object sender, EventArgs e)
{
    int itemNumber = EmployeeListBox.SelectedIndex;
    if (itemNumber > -1)
    {
        EmployeeListBox.Items.RemoveAt(itemNumber);
        WriteEmpsToFile();
    }
    else
    {
        MessageBox.Show("Please select an employee to remove");
    }
}

private void DisplayButton_Click(object sender, EventArgs e)
{
    ReadEmpsFromFile();
}

private void ReadEmpsFromFile()
{
    //check to see if file exist
    if (File.Exists(FILENAME) && new FileInfo(FILENAME).Length > 0)
    {
        //create a pipe from the fileand create translator
        FileStream fs = new FileStream(FILENAME, FileMode.Open);
        BinaryFormatter formatter = new BinaryFormatter();

        //read the entire employee list from the file
        List<Employee> list = formatter.Deserialize(fs) as List<Employee>;
```

```
        //close the pipe
        fs.Close();

        //clear the list box and copy items to the list box
        EmployeeListBox.Items.Clear();
        foreach (Employee emp in list)
        {
            EmployeeListBox.Items.Add(emp);
        }
    }
}

private void PrintPayCheckButton_Click(object sender, EventArgs e)
{
    foreach(Employee emp in EmployeeListBox.Items)
    {
        string line1 = "Pay To: " + emp.EmployeeFirstName + " " + emp.EmployeeLastName;
        string line2 = "Amount of: " + emp.CalcualatePay().ToString("C2");
        string output = "Paycheck:\n\n" + line1 + "\n" + line2;
        MessageBox.Show(output);
    }
}

public void WriteEmpsToFile()
{
    //convert the employee list box to a list
    List<Employee> empList = new List<Employee>();

    foreach (Employee emp in EmployeeListBox.Items)
    {
        empList.Add(emp);
    }

    //open a pipe to the file and create translater
    FileStream fs = new FileStream(FILENAME, FileMode.Create);
    BinaryFormatter formatter = new BinaryFormatter();

    //write the entire employee list to the file
    formatter.Serialize(fs, empList);

    //close the pipe
    fs.Close();



}



private void MainForm_Load(object sender, EventArgs e)
```

```
    {
        ReadEmpsFromFile();
    }

    private void EmployeeListBox_DoubleClick(object sender, EventArgs e)
    {
        //get the selected object
        Employee emp = (Employee)EmployeeListBox.SelectedItem;  //or
EmployeeListBox.SelectedItem as Employee;

        if (emp != null)
        {
            EmployeeInputForm updateForm = new EmployeeInputForm();

            updateForm.Text = "Update Employee Information";
            updateForm.SubmitButton.Text = "Update";
            updateForm.StartPosition = FormStartPosition.CenterParent;
            updateForm.FirstNameTextBox.Text = emp.EmployeeFirstName;
            updateForm.LastNameTextBox.Text = emp.EmployeeLastName;
            updateForm.SSNTextBox.Text = emp.SSN;
            updateForm.DateOfHireTextBox.Text = emp.HireDate.ToShortDateString();
            updateForm.HealthInsTextBox.Text = emp.BenefitsEmp.HealthInsurance;
            updateForm.LifeInsTextBox.Text = emp.BenefitsEmp.LifeInsurance.ToString();
            updateForm.VacationTextBox.Text = emp.BenefitsEmp.Vacation.ToString();

            // check if the employee is a salary or hourly employee
            if (emp is Salary)
            {
                updateForm.HourlyRateLabel.Visible = false;     //hide the hourly controls
                updateForm.HourlyRateTextBox.Visible = false;   //hide the hourly controls
                updateForm.HoursWrokedlabel.Visible = false;    //hide the hourly controls
                updateForm.HoursWorkedtextBox.Visible = false;  //hide the hourly controls
                updateForm.SalaryLabel.Visible = true;          //show the salary controls
                updateForm.SalaryTextBox.Visible = true;        //show the salary controls

                //set the radio button
                updateForm.SalaryRadioButton.Checked = true;

                //Cast the employee object to a salary object
                Salary sal = (Salary)emp;

                //show the salary attribute
                updateForm.SalaryTextBox.Text = sal.AnnualSalary.ToString("F2");

            }

            else if (emp is Hourly)
            {
                updateForm.HourlyRateLabel.Visible = true;      //show the hourly controls
                updateForm.HourlyRateTextBox.Visible = true;    //show the hourly controls
                updateForm.HoursWrokedlabel.Visible = true;     //show the hourly controls
```

```
    updateForm.HoursWorkedtextBox.Visible = true;   //show the hourly controls
    updateForm.SalaryLabel.Visible = false;         //hide the salary controls
    updateForm.SalaryTextBox.Visible = false;       //hide the salary controls

    //set the radio button
    updateForm.HourlyRadioButton.Checked = true;

    //Cast the employee object to a hourly object
    Hourly hrly = (Hourly)emp;

    //show the hourly attributes
    updateForm.HourlyRateTextBox.Text = hrly.HourlyRate.ToString("F2");
    updateForm.HoursWorkedtextBox.Text = hrly.HoursWorked.ToString("F1");
}
else
{
    MessageBox.Show("Employee type not found");
    return;
}

DialogResult result = updateForm.ShowDialog();

//if caceled stop method
if (result == DialogResult.Cancel)
    return; // <== ends method since canceled

//delete selected object
int position = EmployeeListBox.SelectedIndex;
EmployeeListBox.Items.RemoveAt(position);

//create new employee from 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.DateOfHireTextBox.Text);
string healthInsurance = updateForm.HealthInsTextBox.Text;
int lifeInsurance = int.Parse(updateForm.LifeInsTextBox.Text);
int vacation = int.Parse(updateForm.VacationTextBox.Text);
Benefits benifits = new Benefits(healthInsurance, lifeInsurance, vacation);

if (updateForm.SalaryRadioButton.Checked)
{
    double salary = double.Parse(updateForm.SalaryTextBox.Text);
    newEmp = new Salary(fName, lName, ssn, hireDate, benifits, 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, hireDate, benifits, hourlyRate,
hoursWorked);
            }
            else
            {
                MessageBox.Show("Please select a employee type");
                return;
            }

            //add new employee to list box
            EmployeeListBox.Items.Add(newEmp);

            // update the data file
            WriteEmpsToFile();


        }
        //create a new form
    }
  }//closes the class
}//closses the name space
```

Inputform.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Venegas_CourseProject_Part2
{
  public partial class EmployeeInputForm : Form
  {
    public EmployeeInputForm()
    {
      InitializeComponent();
    }

    private void SubmitButton_Click(object sender, EventArgs e)
    {
      this.DialogResult = DialogResult.OK;
      this.Hide();
    }

    private void ExitButton_Click(object sender, EventArgs e)
```

```
        {
            this.DialogResult = DialogResult.Cancel;
            this.Hide();
        }

        private void HourlyradioButton_CheckedChanged(object sender, EventArgs e)
        {
            if (HourlyRadioButton.Checked)
            {
                //show hourly controls
                HourlyRateLabel.Visible = true;
                HoursWrokedlabel.Visible = true;
                HourlyRateTextBox.Visible = true;
                HoursWorkedtextBox.Visible = true;

                //hide salary controls
                SalaryLabel.Visible = false;
                SalaryTextBox.Visible = false;

            }
        }

        private void SalaryradioButton_CheckedChanged(object sender, EventArgs e)
        {
            if(SalaryRadioButton.Checked)
            {
                //show salary controls
                SalaryLabel.Visible = true;
                SalaryTextBox.Visible = true;

                //hide hourly controls
                HourlyRateLabel.Visible = false;
                HoursWrokedlabel.Visible = false;
                HourlyRateTextBox.Visible = false;
                HoursWorkedtextBox.Visible = false;
            }
        }
    }
}
```

Employee.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace Venegas_CourseProject_Part2
```

```
{
   [Serializable]
   public abstract class Employee
   {
      //attributes
      protected string employeeFirstName;
      protected string employeeLastName;
      protected string ssn;
      protected DateTime hireDate;
      protected Benefits benefits;


      //default constructor
      public Employee()
      {
         employeeFirstName = "John";
         employeeLastName = "Doe";
         ssn = "123-45-6789";
         hireDate = DateTime.Today;
         benefits = new Benefits();
      }


      //parameterized constructor
      public Employee(string employeeFirstName, string employeeLastName, string ssn,
DateTime hireDate, Benefits benefits)
      {
         this.employeeFirstName = employeeFirstName;
         this.employeeLastName = employeeLastName;
         this.ssn = ssn;
         this.hireDate = hireDate;
         this.benefits = benefits;
      }

      //behaviors

      public override string ToString()
      {
         return "firstName= " + employeeFirstName + ", lastName= " + employeeLastName +
            ", ssn= " + ssn + ", hireDate= " + hireDate.ToShortDateString()
            + ", healthIns= " + benefits.HealthInsurance + ", lifeIns= " + benefits.LifeInsurance + ",
vacation= " + benefits.Vacation;
      }

      public abstract Double CalcualatePay();

      //properties
      public string EmployeeFirstName
      {
         get
         {
```

```
        return employeeFirstName;
    }
    set
    {
        employeeFirstName = value;
    }
}

public string EmployeeLastName
{
    get
    {
        return employeeLastName;
    }
    set
    {
        employeeLastName = value;
    }
}

public DateTime HireDate
{
    get
    {
        return hireDate;
    }
    set
    {
        hireDate = value;
    }
}

public string SSN
{
    get
    {
        return ssn;
    }
    set
    {
        ssn = value;
    }
}

public Benefits BenefitsEmp
{
    get
    {
        return benefits;
    }
    set
```

```
        {
            benefits = value;
        }
    }

  }
}
```

Benefits.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Venegas_CourseProject_Part2
{
   [Serializable]
   public class Benefits
   {
      //attributes

      private string healthInsurance;
      private int lifeInsurance;
      private int vacation;

      //default constructor
      public Benefits()
      {
         healthInsurance = "Refused";
         lifeInsurance = 0;
         vacation = 0;
      }


      //constructor with parameters

      public Benefits(string healthInsurance, int lifeInsurance, int vacation)
      {
         this.healthInsurance = healthInsurance;
         this.lifeInsurance = lifeInsurance;
         this.vacation = vacation;
      }

      //behaviors
      public override string ToString()
      {
         return "Health Insurance: " + healthInsurance + "\nLife Insurance: " +
lifeInsurance.ToString("C0") + "\nVacation: " + vacation;
      }
```

```
        //public string (properties)
        public string HealthInsurance
        {
            get { return healthInsurance; }
            set { healthInsurance = value; }
        }

        public int LifeInsurance
        {
            get { return lifeInsurance; }
            set { lifeInsurance = value; }
        }

        public int Vacation
        {
            get { return vacation; }
            set { vacation = value; }
        }


    }
}
```

Hourly.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Venegas_CourseProject_Part2
{
    [Serializable]
    public class Hourly : Employee
    {
        //attributes
        protected double hourlyRate;
        protected double hoursWorked;

        //default constructor
        public Hourly() : base()
        {
            hourlyRate = 0.0;
            hoursWorked = 0.0;
        }

        //parameterized constructor
```

```
        public Hourly(string employeeFirstName, string employeeLastName, string ssn, DateTime
hireDate, Benefits benefits, double hourlyRate, double hoursWorked) :
base(employeeFirstName, employeeLastName, ssn, hireDate, benefits)
        {
            this.hourlyRate = hourlyRate;
            this.hoursWorked = hoursWorked;
        }

        //behaviors
        public override double CalcualatePay()
        {
            double pay = 0.0;
            if (hoursWorked > 40)
            {
                double basepay = 40 * hourlyRate;
                double overtime = (hoursWorked - 40) * hourlyRate * 1.5;
                pay = basepay + overtime;
            }
            else
            {
                pay = hoursWorked * hourlyRate;
            }
            return pay;
        }
        //properties
        public double HourlyRate
        {
            get { return hourlyRate; }
            set { hourlyRate = value; }
        }
        public double HoursWorked
        {
            get { return hoursWorked; }
            set { hoursWorked = value; }
        }
        //toString
        public override string ToString()
        {
            return base.ToString() + ", hourlyRate= " + hourlyRate.ToString("C2") + ", hoursWorked=
" + hoursWorked.ToString("C2");
        }
        //validate
        public string Validate()
        {
            string result = "";
            if (hourlyRate < 0)
            {
                result += "Hourly rate cannot be negative\n";
            }
            if (hoursWorked < 0)
```

```
            {
                result += "Hours worked cannot be negative\n";
            }
            return result;
        }
    }
}
```

Salary.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Venegas_CourseProject_Part2
{
    [Serializable]
    public class Salary : Employee
    {
        //attributes
        protected double annualSalary;

        //constructors
        public Salary() : base()
        {
            annualSalary = 0.0;
        }

        public Salary(string employeeFirstName, string employeeLastName, string ssn,
            DateTime hireDate, Benefits benefits, double annualSalary) : base(employeeFirstName,
employeeLastName, ssn, hireDate, benefits)
        {
            this.annualSalary = annualSalary;
        }
        //behaviors
        public override double CalcualatePay()
        {
            return annualSalary / 26.0;     //double result = 9/10;
        }

        public override string ToString()
        {
            return base.ToString() + ", annualSalary= " + annualSalary.ToString("C2");
        }

        //properties
        public double AnnualSalary
        {
            get { return annualSalary; }
```

```
            set { annualSalary = value; }
        }
    }
}
```