**Program**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace C# 4

{

classProgram

{

staticvoid Main(string[] args)

{

ApplicationUtilities.DisplayApplicationInformation();

ApplicationUtilities.DisplayDivider("Start Program");

ApplicationUtilities.DisplayDivider("Prompt for Employee Information and create first employee");

Employee employee1 = new Employee();

employee1.FirstName = InputUtilities.getStringInputValue("First Name");

employee1.LastName = InputUtilities.getStringInputValue("Last Name");

employee1.Gender = InputUtilities.getCharInputValue("Gender");

employee1.Dependents = InputUtilities.getIntegerInputValue("Dependants");

employee1.AnnualSalary = InputUtilities.getDoubleInputValue("Annual Salary");

employee1.Employee\_Benefit.Health\_Info = InputUtilities.getStringInputValue("Health Insurance Company");

employee1.Employee\_Benefit.Life\_Info = InputUtilities.getDoubleInputValue("Insurance Amount");

employee1.Employee\_Benefit.Vac\_Days = InputUtilities.getIntegerInputValue("# Vacations Days");

Console.WriteLine(employee1.ToString());

//Updated weekly pay

ApplicationUtilities.DisplayDivider("Get Weekly Pay");

employee1.AnnualSalary = InputUtilities.getDoubleInputValue("Updated Annual Salary");

string thename = employee1.FirstName + " " + employee1.LastName;

Console.WriteLine(thename + " modified weekly pay: " + employee1.CalculateWeeklyPay(employee1.AnnualSalary).ToString("C2"));

ApplicationUtilities.PauseExecution();

ApplicationUtilities.DisplayDivider("Number of Employee Object(s):" + Employee.NumberEmployees);

//display employee2 information

Benefits emp2Ben = new Benefits("Tricare", 250000, 21);

Employee employee2 = new Employee("Mary", "Noia", 'F', 3, 100000, emp2Ben);

Console.WriteLine(employee2.ToString());

ApplicationUtilities.DisplayDivider("Number of Employee Object(s):" + Employee.NumberEmployees);

//display employee3 inforamtion

Benefits emp3Ben = new Benefits("", 1000000, -10);

Employee employee3 = new Employee("Sue", "Smith", 'F', 3, 100000, emp3Ben);

Console.WriteLine(employee3.ToString());

ApplicationUtilities.DisplayDivider("Number of Employee Object(s):" + Employee.NumberEmployees);

ApplicationUtilities.TerminateApplication();

}

}

}

**Application Utilities**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace C# 4B

{

publicclassApplicationUtilities

{

publicstaticvoid DisplayApplicationInformation()

{

Console.WriteLine("Welcome the Basic Employee Program");

Console.WriteLine("CIS247A, Week 4 Lab");

Console.WriteLine("Name: Kevin Nguyen");

Console.WriteLine("This program accepts user input as a string, then makes the \nappropriate data conversion and assigns the value to Employee objects and \n collects and converse the employee information.");

Console.WriteLine();

}

publicstaticvoid DisplayDivider(string outputTitle)

{

Console.WriteLine("\n\*\*\*\*\*\*\*\*\* " + outputTitle + " \*\*\*\*\*\*\*\*\*\n");

}

publicstaticvoid TerminateApplication()

{

DisplayDivider("Program Termination");

Console.Write("Thank you. Press any key to terminate the program...");

Console.ReadLine();

}

publicstaticvoid PauseExecution()

{

Console.Write("\nProgram paused, press any key to continue...");

Console.ReadLine();

Console.WriteLine();

}

}

}

**Input Utilities**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace C# 4C

{

publicclassInputUtilities

{

publicstaticstring GetInput(string inputType)

{

string strInput = String.Empty;

Console.Write("Enter the " + inputType + ": ");

strInput = Console.ReadLine();

return strInput;

}

publicstaticstring getStringInputValue(string inputType)

{

string value = String.Empty;

bool valid = false;

string inputString = String.Empty;

do

{

inputString = GetInput(inputType);

if (!String.IsNullOrEmpty(inputString))

{

value = inputString;

valid = true;

}

else

{

value = "Invalid input";

valid = false;

}

if (!valid)

Console.WriteLine("Invalid " + inputType + " try again!");

} while (!valid);

return value;

}

publicstaticint getIntegerInputValue(string inputType)

{

bool valid = false;

int value = 0;

string inputString = String.Empty;

do

{

inputString = GetInput(inputType);

if (!(String.IsNullOrEmpty(inputString)))

{

valid = Int32.TryParse(inputString, out value);

}

if (!valid)

Console.WriteLine("Invalid " + inputType + " try again!");

} while (!valid);

return value;

}

publicstaticdouble getDoubleInputValue(string inputType)

{

bool valid = false;

double value = 0;

string inputString = String.Empty;

do

{

inputString = GetInput(inputType);

if (!(String.IsNullOrEmpty(inputString)))

{

valid = Double.TryParse(inputString, out value);

}

if (!valid)

Console.WriteLine("Invalid " + inputType + " try again!");

} while (!valid);

return value;

}

publicstaticchar getCharInputValue(string inputType)

{

bool valid = false;

char value = 'u';

string inputString = String.Empty;

do

{

inputString = GetInput(inputType);

if (!(String.IsNullOrEmpty(inputString)))

{

valid = Char.TryParse(inputString, out value);

}

if (!valid)

Console.WriteLine("Invalid " + inputType + " try again!");

} while (!valid);

return value;

}

}

}

**Employee**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace C# 4D

{

classEmployee

{

privateconstint MIN\_DEPENDENTS = 0;

privateconstint MAX\_DEPENDENTS = 10;

privateconstdouble MIN\_SALARY = 20000;

privateconstdouble MAX\_SALARY = 100000;

privateconststring DEFAULT\_NAME = "Not Given";

privateconstchar DEFAULT\_GENDER = 'U';

// the variables are declared

protectedstring firstName = DEFAULT\_NAME, lastName = DEFAULT\_NAME;

protectedchar gender = DEFAULT\_GENDER;

protectedint dependents = MIN\_DEPENDENTS;

protecteddouble annualSalary = MIN\_SALARY;

protectedstaticint numEmployees = 0;

protected Benefits benefits;

// default constructor - you don't expect the user to put anything in.

public Employee()

{

firstName = DEFAULT\_NAME;

lastName = DEFAULT\_NAME;

gender = DEFAULT\_GENDER;

dependents = MIN\_DEPENDENTS;

annualSalary = MIN\_SALARY;

benefits = new Benefits();

}

// implement the overloaded constructor

public Employee(string fn, string ln, char gen, int dep, double ans, Benefits ben)

{

firstName = fn;

lastName = ln;

gender = gen;

dependents = dep;

annualSalary = ans;

benefits = ben;

}

// we calculate the weekly pay (52 weeks in a year)

publicdouble CalculateWeeklyPay()

{

return annualSalary / 52;

}

publicdouble CalculateWeeklyPay(double modifiedSalary)

{

return modifiedSalary / 52;

}

// place the override tostring

publicoverridestring ToString()

{

string empvar = "============= Employee Information =============";

empvar += "\nName:" + firstName + " " + lastName;

empvar += "\nGender: " + Char.ToUpper(gender);

empvar += "\nDependants: " + dependents;

empvar += "\nAnnualSalary: " + annualSalary.ToString("C2");

empvar += "\nWeeklyPay: " + CalculateWeeklyPay().ToString("C2");

empvar += benefits.ToString();

return empvar;

}

publicstring FirstName

{

get { return firstName; }

set

{

if (value.Trim() == String.Empty || value.Trim() == "")

firstName = DEFAULT\_NAME;

else

firstName = value;

}

}

publicstring LastName

{

get { return lastName; }

set

{

if (value.Trim() == String.Empty || value.Trim() == "")

lastName = DEFAULT\_NAME;

else

lastName = value;

}

}

publicchar Gender

{

get { return gender; }

set

{

if (Char.ToUpper(value) == 'F' || Char.ToUpper(value) == 'M')

gender = value;

else

gender = DEFAULT\_GENDER;

}

}

publicint Dependents

{

get { return dependents; }

set

{

if (value< MIN\_DEPENDENTS)

dependents = MIN\_DEPENDENTS;

elseif (value> MAX\_DEPENDENTS)

dependents = MAX\_DEPENDENTS;

else

dependents = value;

}

}

publicdouble AnnualSalary

{

get { return annualSalary; }

set

{

if (value< MIN\_SALARY)

annualSalary = MIN\_SALARY;

elseif (value> MAX\_SALARY)

annualSalary = MAX\_SALARY;

else

annualSalary = value;

}

}

publicstaticint NumberEmployees

{

get

{

return numEmployees += 1;

}

}

public Benefits Employee\_Benefit

{

get { return benefits; }

set

{

if (benefits != null)

benefits = new Benefits();

else

benefits = value;

}

}

~Employee() { }

}

}

**Benefits**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace C# 4E

{

classBenefits

{

privateconststring DEFAULT\_HEALTH\_INSURANCE = "Blue Cross";

privateconstdouble MIN\_LIFE\_INSURANCE = 0;

privateconstdouble MAX\_LIFE\_INSURANCE = 1000000;

privateconstint MIN\_VACATION = 0;

privateconstint MAX\_VACATION = 45;

// the variables are declared

privatestring healthInsuranceCompany = DEFAULT\_HEALTH\_INSURANCE;

privatedouble lifeInsuranceAmount = MIN\_LIFE\_INSURANCE;

privateint vacationDays = MIN\_VACATION;

// default constructor - you don't expect the user to put anything in.

public Benefits()

{

healthInsuranceCompany = DEFAULT\_HEALTH\_INSURANCE;

lifeInsuranceAmount = MIN\_LIFE\_INSURANCE;

vacationDays = MIN\_VACATION;

}

public Benefits(string health, double life, int vacation)

{

healthInsuranceCompany = health;

lifeInsuranceAmount = life;

vacationDays = vacation;

}

publicoverridestring ToString()

{

string empvar = "\n============= Benefits Information =============";

empvar += "\nhealthInsurancCompany:" + healthInsuranceCompany;

empvar += "\nlifeInsuranceAmount: " + lifeInsuranceAmount.ToString("C2");

empvar += "\nvacationDays: " + vacationDays;

return empvar;

}

publicstring Health\_Info

{

get { return healthInsuranceCompany; }

set

{

if (value.Trim() == String.Empty || value.Trim() == "")

healthInsuranceCompany = DEFAULT\_HEALTH\_INSURANCE;

else

healthInsuranceCompany = value;

}

}

publicdouble Life\_Info

{

get { return lifeInsuranceAmount; }

set

{

if (value< MIN\_LIFE\_INSURANCE)

lifeInsuranceAmount = MIN\_LIFE\_INSURANCE;

elseif (value> MAX\_LIFE\_INSURANCE)

lifeInsuranceAmount = MAX\_LIFE\_INSURANCE;

else

lifeInsuranceAmount = value;

}

}

publicint Vac\_Days

{

get { return vacationDays; }

set

{

if (value< MIN\_VACATION)

vacationDays = MIN\_VACATION;

elseif (value> MAX\_VACATION)

vacationDays = MAX\_VACATION;

else

vacationDays = value;

}

}

~Benefits() { }

}

}