**Benefits**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

namespace C#5

{

publicclassBenefits

{

privateconststring DEFAULT\_HEALTH\_INSURANCE = "not given";

privateconstdouble MIN\_LIFE\_INSURANCE = 0;

privateconstdouble MAX\_LIFE\_INSURANCE = 1000000;

privateconstint MIN\_VACATION = 0;

privateconstint MAX\_VACATION = 45;

privatestringhealthInsuranceCompany;

privatedoublelifeInsuranceAmount;

privateintvacationDays;

#region Constructors

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;

}

#endregion

#region Properties

publicstringHealthInsuranceCompany

{

get

{

returnhealthInsuranceCompany;

}

set

{

if (String.IsNullOrEmpty(value))

healthInsuranceCompany = DEFAULT\_HEALTH\_INSURANCE;

else

healthInsuranceCompany = value;

}

}

publicdoubleLifeInsuranceAmount

{

get

{

returnlifeInsuranceAmount;

}

set

{

if (value>= MIN\_LIFE\_INSURANCE &&value<= MAX\_LIFE\_INSURANCE)

lifeInsuranceAmount = value;

elseif (value< MIN\_LIFE\_INSURANCE)

lifeInsuranceAmount = MAX\_LIFE\_INSURANCE;

else

lifeInsuranceAmount = MAX\_LIFE\_INSURANCE;

}

}

publicintVacationDays

{

get

{

returnvacationDays;

}

set

{

if (value>= MIN\_VACATION &&value<= MAX\_VACATION)

vacationDays = value;

elseif (value< MIN\_VACATION)

vacationDays = MIN\_VACATION;

else

vacationDays = MAX\_VACATION;

}

}

#endregion

#region Class Methods

publicoverridestringToString()

{

string output;

output = "\n\t============ Benefit Information ============";

output += "\n\tHealth Insurance Company: " + HealthInsuranceCompany;

output += "\n\t Life Insurance Amount: " + LifeInsuranceAmount.ToString("C2");

output += "\n\t Vacation Days: " + VacationDays;

return output;

}

#endregion

}

}

**Employee**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

namespace C#5B

{

publicclassEmployee

{

publicconstdouble MIN\_SALARY = 20000;

publicconstdouble MAX\_SALARY = 100000;

publicconstint MIN\_DEPENDENTS = 0;

publicconstint MAX\_DEPENDENTS = 10;

publicconststring DEFAULT\_NAME = "not given";

protectedstringfirstName;

protectedstringlastName;

protecteddoubleannualSalary;

protectedchar gender;

protectedint dependents;

protected Benefits benefit;

protectedstringemployeeType;

//since static, there will be only one copy of the number created variable

protectedstaticintnumberEmployees = 0;

public Employee()

{

FirstName = String.Empty;

LastName = String.Empty;

AnnualSalary = MIN\_SALARY;

Dependents = MIN\_DEPENDENTS;

Benefit = newBenefits();

employeeType = "Generic";

//make sure to increment number of employees each time a instance is created

numberEmployees += 1;

}

public Employee(stringfirstname, stringlastname, char gender, int dependents, double salary, Benefits benefit, stringEmployeeType)

{

FirstName = firstname;

LastName = lastname;

AnnualSalary = salary;

Gender = gender;

Dependents = dependents;

Benefit = benefit;

EmployeeType = employeeType;

//make sure to increment number of employees each time a instance is created

numberEmployees += 1;

}

publicstaticintNumberOfEmployees

{

get { returnnumberEmployees; }

}

public Benefits Benefit

{

get { return benefit; }

set

{

if (value != null)

benefit = value;

else

benefit = new Benefits();

}

}

publicstringFirstName

{

get { returnfirstName; }

set

{

if (String.IsNullOrEmpty(value))

firstName = DEFAULT\_NAME;

else

firstName = value;

}

}

publicstringLastName

{

get { returnlastName; }

set

{

if (String.IsNullOrEmpty(value))

lastName = DEFAULT\_NAME;

else

lastName = value;

}

}

publicchar Gender

{

get { return gender; }

set

{

char input;

input = Char.ToUpper(value);

if (input == 'F' || input == 'M')

gender = input;

else

gender = 'U';

}

}

publicstringFullGender

{

//create a read only property to

//provide the full text of the gender

get

{

string value;

switch (Gender)

{

case'F':

value = "Female";

break;

case'M':

value = "Male";

break;

default:

value = "Unknown";

break;

}

return value;

}

}

publicint Dependents

{

get { return dependents; }

set

{

if (value>= MIN\_DEPENDENTS &&value<= MAX\_DEPENDENTS)

dependents = value;

elseif (value< MIN\_DEPENDENTS)

dependents = MIN\_DEPENDENTS;

else

dependents = MAX\_DEPENDENTS;

}

}

publicdoubleAnnualSalary

{

get { returnannualSalary; }

set

{

if (value< MIN\_SALARY)

annualSalary = MIN\_SALARY;

elseif (value> MAX\_SALARY)

annualSalary = MAX\_SALARY;

else

annualSalary = value;

}

}

publicstringEmployeeName

{

get { returnFirstName + " " + LastName; }

}

publicstringEmployeeType

{

get { returnemployeeType; }

}

public Employee(stringemployeeType) : this() { }

publicvirtualdoubleCalculateWeeklyPay()

{

returnAnnualSalary / 52;

}

publicvirtualdoubleCalculateWeeklyPay(doublemodifiedSalary)

{

AnnualSalary = modifiedSalary;

returnCalculateWeeklyPay();

}

publicoverridestringToString()

{

string output;

output = "\n============ Employee Information ============";

output += "\n\t Type:\t" + employeeType;

output += "\n\t Name:\t" + firstName + " " + lastName;

output += "\n\t Gender:\t" + FullGender;

output += "\n\t Dependents:\t" + Dependents;

output += "\n\tAnnual Salary:\t" + AnnualSalary.ToString("C2");

output += "\n\t Weekly Pay:\t" + CalculateWeeklyPay().ToString("C2");

output += Benefit.ToString();

return output;

}

}

}

**Hourly**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace C#5C

{

classHourly : Employee

{

protectedconstdouble MIN\_WAGE = 10;

protectedconstdouble MAX\_WAGE = 75;

protectedconstdouble MIN\_HOURS = 0;

protectedconstdouble MAX\_HOURS = 50;

protectedconststring TEMPORARY = "temporary";

protectedconststring PART\_TIME = "part time";

protectedconststring FULL\_TIME = "full time";

protectedconststring INVALID\_CATEGORY = "invalid category";

protecteddouble wage;

protecteddouble hours;

protectedstring category;

publicdouble Wage

{

get { return wage; }

set

{

if (value< MIN\_WAGE)

wage = MIN\_WAGE;

else

if (value> MAX\_WAGE)

wage = MAX\_WAGE;

else

wage = value;

base.AnnualSalary = CalculateWeeklyPay() \* 48;

}

}

publicdouble Hours

{

get { return hours; }

set

{

if (value< MIN\_HOURS)

hours = MIN\_HOURS;

else

if (value> MAX\_HOURS)

hours = MAX\_HOURS;

else

hours = value;

base.AnnualSalary = CalculateWeeklyPay() \* 48;

}

}

publicstring Category

{

get { return category; }

set

{

if (value == FULL\_TIME || value == PART\_TIME || value == TEMPORARY)

category = value;

else

category = INVALID\_CATEGORY;

}

}

publicoverridedoubleCalculateWeeklyPay()

{

return wage \* hours;

}

public Hourly()

{

employeeType = "Hourly";

}

publicoverridestringToString()

{

string output;

output = "============ Employee Information ============";

output += "\n\t Name:\t" + firstName + " " + lastName;

output += "\n\t Gender:\t" + gender;

output += "\n\t Dependents:\t" + dependents;

output += "\n\ttAnnual Salary:\t" + annualSalary.ToString("C2");

output += "\n\t Weekly Pay:\t" + CalculateWeeklyPay().ToString("C2");

output += "\n\t Employee Type:\t" + employeeType;

output += benefit.ToString();

output += "\n\t Category:\t" + category;

output += "\n\t Hours:\t" + hours;

output += "\n\t Wages:\t" + wage;

return output;

}

public Hourly(stringfname, stringlname, char gen, intdep, doublesal, Benefits ben, stringempType, string cat)

: base(fname, lname, gen, dep, sal, ben, empType)

{

category = cat;

numberEmployees += 1;

}

}

}

**Salaried**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace C#5D

{

classSalaried : Employee

{

protectedconstint MIN\_MANAGEMENT\_LEVEL = 0;

protectedconstint MAX\_MANAGEMENT\_LEVEL = 10;

protectedconstdouble BONUS\_PERCENT = .10;

protectedintmanagementLevel;

publicintManagementLevel

{

get { returnmanagementLevel; }

set

{

if (value< MIN\_MANAGEMENT\_LEVEL)

managementLevel = MIN\_MANAGEMENT\_LEVEL;

else

if (value> MAX\_MANAGEMENT\_LEVEL)

managementLevel = MAX\_MANAGEMENT\_LEVEL;

else

managementLevel = value;

}

}

public Salaried()

{

employeeType = "Salaried";

}

public Salaried(stringfname, stringlname, char gen, intdep, doublesal, Benefits ben, stringempType, intmanLevel)

: base(fname, lname, gen, dep, sal, ben, empType)

{

managementLevel = manLevel;

numberEmployees += 1;

}

publicoverridedoubleCalculateWeeklyPay()

{

returnannualSalary / 52 \* (1 + managementLevel \* BONUS\_PERCENT);

}

publicoverridestringToString()

{

string output;

output = "============ Employee Information ============";

output += "\n\t Name:\t" + firstName + " " + lastName;

output += "\n\t Gender:\t" + gender;

output += "\n\t Dependents:\t" + dependents;

output += "\n\t Annual Salary:\t" + annualSalary.ToString("C2");

output += "\n\t Weekly Pay:\t" + CalculateWeeklyPay().ToString("C2");

output += "\n\t Employee Type:\t" + employeeType;

output += benefit.ToString();

output += "\n\tManagement Level:\t" + managementLevel;

return output;

}

}

}

**Employee Input**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace C#5E

{

classEmployeeInput

{

publicstaticvoidCollectEmployeeInformation(Employee theEmployee)

{

theEmployee.FirstName = InputUtilities.GetStringInputValue("First Name");

theEmployee.LastName = InputUtilities.GetStringInputValue("Last Name");

theEmployee.Gender = InputUtilities.GetCharInputValue("gender");

theEmployee.Dependents = InputUtilities.GetIntegerInputValue("Number of Dependents");

//if (theEmployee.EmployeeType == "Salaried")

// theEmployee.AnnualSalary = InputUtilities.GetDoubleInputValue("Annual Salary");

theEmployee.Benefit.HealthInsuranceCompany = InputUtilities.GetStringInputValue("Insurance Company");

theEmployee.Benefit.LifeInsuranceAmount = InputUtilities.GetDoubleInputValue("Insurance Amount");

theEmployee.Benefit.VacationDays = InputUtilities.GetIntegerInputValue("Number of Vacation Days");

}

publicstatic

voidCollectHourlyInformation(Hourly theEmployee)

{

theEmployee.Wage = InputUtilities.GetDoubleInputValue("Wage");

theEmployee.Hours = InputUtilities.GetDoubleInputValue("Hours");

theEmployee.Category = InputUtilities.GetStringInputValue("Category");

}

publicstaticvoidCollectSalariedInformation(Salaried theEmployee)

{

theEmployee.ManagementLevel = InputUtilities.GetIntegerInputValue("Management Level");

theEmployee.AnnualSalary = InputUtilities.GetDoubleInputValue("Annual Salary");

}

}

}

**Employee Output**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace C#5F

{

classEmployeeOutput

{

publicstaticvoidDisplayEmployeeInformation(Employee theEmployee)

{

Console.WriteLine(theEmployee.ToString());

}

publicstaticvoidDisplayNumberObject()

{

Console.WriteLine("Number of Employees created: " + Employee.NumberOfEmployees);

}

}

}

**Input Utilities**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

namespace C#5G

{

publicclassInputUtilities

{

privatestaticstringGetInput(stringinputType)

{

stringstrInput = String.Empty;

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

strInput = Console.ReadLine();

returnstrInput;

}

publicstaticstringGetStringInputValue(stringinputType)

{

string value = String.Empty;

bool valid = false;

stringinputString = 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;

}

publicstaticintGetIntegerInputValue(stringinputType)

{

bool valid = false;

int value = 0;

stringinputString = 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;

}

publicstaticdoubleGetDoubleInputValue(stringinputType)

{

bool valid = false;

double value = 0;

stringinputString = 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;

}

publicstaticcharGetCharInputValue(stringinputType)

{

bool valid = false;

char value = 'u';

stringinputString = 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;

}

}

}

**Program**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

namespace C#5H

{

classProgram

{

staticvoid Main(string[] args)

{

ApplicationUtilities.DisplayApplicationInformation();

Employee[] emps = new Employee[3];

emps[0] = new Employee("Generic");

emps[1] = new Hourly();

emps[2] = new Salaried();

for (inti = 0; i<emps.Length; i++)

{

ApplicationUtilities.DisplayDivider("Employee Information");

EmployeeInput.CollectEmployeeInformation(emps[i]);

if (emps[i] is Hourly)

{

EmployeeInput.CollectHourlyInformation((Hourly)emps[i]);

}

elseif (emps[i] is Salaried)

{

EmployeeInput.CollectSalariedInformation((Salaried)emps[i]);

}

EmployeeOutput.DisplayEmployeeInformation(emps[i]);

ApplicationUtilities.PauseExecution();

}

EmployeeOutput.DisplayNumberObject();

ApplicationUtilities.TerminateApplication();

}

}

}

**Application Utilities**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

namespace C#5I

{

publicclassApplicationUtilities

{

publicstaticvoidDisplayApplicationInformation()

{

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

Console.WriteLine("CIS247A, Week 5 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"

+ "\ncollects and converts the employee benefit information.");

Console.WriteLine();

}

publicstaticvoidDisplayDivider(stringoutputTitle)

{

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

}

publicstaticvoidTerminateApplication()

{

DisplayDivider("Program Termination");

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

Console.ReadLine();

}

publicstaticvoidPauseExecution()

{

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

Console.ReadLine();

Console.WriteLine();

}

}

}