/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* IComparable: Part 7

\*

\* Core Topics:

\* 1. Implementing IComparable interface.

\* 2. Required to use for Sort methods of arrays

\* and any type of collection.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

usingSystem**;**

usingSystem.Collections.Generic**;**

namespaceIComparableInterface

**{**

internalclassEmployees

**{**

internalreadonlyintNumberOfEmployees**;**

privateList<Employee>employees**;**

internalEmployees**(**intnumberOfEmployees**)**

**{**

NumberOfEmployees=numberOfEmployees**;**

employees=newList<Employee>**(**NumberOfEmployees**);**

**}**

internalEmployeethis**[**intindex**]**

**{**

get

**{**

Employeeemp=null**;**

if **(**index>=0&&index<employees.Count**)**

**{**

emp=employees**[**index**];**

**}**

returnemp**;**

**}**

privateset

**{**

employees.Add**(**value**);**

**}**

**}**

internalvoidPrintAllEmployees**()**

**{**

Console.WriteLine**(**

"\n\n\n\*\*\* Printing Basic Info for All Employees \*\*\*"**);**

for **(**inti=0**;** i<NumberOfEmployees**;** i++**)**

**{**

Console.WriteLine**(**"\n\nID No. {0}:{1}"**,** i+1**,**

employees**[**i**]**.ToString**());**

**}**

**}**

internalvoidAddEmployee**(**Employeee**)**

**{**

if **(**employees.Count<employees.Capacity**)**

**{**

this**[**employees.Count**]** =e**;**

**}**

**}**

internalvoidSortEmployees**()**

**{**

**employees.Sort();**

Console.WriteLine**(**

"\nAll employees are sorted by last name, first name."**);**

**}**

**}**

internalabstractclassEmployee **: IComparable**

**{**

privatestring\_firstName**;**

privatestring\_lastName**;**

privatestring\_socialSecurityNumber**;**

privatestring\_employeeType**;**

publicEmployee**(**stringfirstName**,** stringlastName**,** stringssn**,**

stringemployeeType**)**

**{**

FirstName=firstName**;**

LastName=lastName**;**

SocialSecurityNumber=ssn**;**

EmployeeType=employeeType**;**

**}**

**int System.IComparable.CompareTo(object o)**

**{**

intresult=1**;**

// Using the 'as' keyword, attempt to convert the object passed

// in to an Employee object. This ensures that the object

// passed in is comparable to the current object.

**Employee emp = o as Employee;**

if **(**null!=emp**)**

**{**

// The properties FirstName and LastName return a string. The

// System.String type supports a CompareTo() method as well

// so we'll just use it to perform the comparison.

**result =**

**this.FullName.ToUpper().CompareTo(emp.FullName.ToUpper());**

**}**

else

**{**

Console.WriteLine**(**"\nEMPLOYEE: Object passed is in "+

"not an Employee object - cannot compare."**);**

**}**

returnresult**;**

**}**

publicstringFirstName

**{**

get **{** return\_firstName**; }**

protectedset **{** \_firstName=value**; }**

**}**

publicstringLastName

**{**

get **{** return\_lastName**; }**

protectedset **{** \_lastName=value**; }**

**}**

publicstringFullName

**{**

get **{** return\_lastName+" "+\_firstName**; }**

**}**

publicstringSocialSecurityNumber

**{**

get **{** return\_socialSecurityNumber**; }**

protectedset **{** \_socialSecurityNumber=value**; }**

**}**

publicstringEmployeeType

**{**

get **{** return\_employeeType**; }**

protectedset **{** \_employeeType=value**; }**

**}**

publicabstractdecimalPayCheck**();**

publicoverridestringToString**()**

**{**

stringoutput="\n\tEmployee Type: "+EmployeeType+

"\n\tLast name: "+LastName+

"\n\tFirst name: "+FirstName+

"\n\tSSN: "+SocialSecurityNumber**;**

returnoutput**;**

**}**

**}**

internalclassSalariedEmployee **:** Employee

**{**

privatedecimal\_weeklySalary**;**

publicSalariedEmployee**(**stringfirstName**,** stringlastName**,**

stringssn**,** decimalweeklySalary**)**

**:** this**(**firstName**,** lastName**,** ssn**,** weeklySalary**,**

"Salaried Employee"**)**

**{ }**

publicSalariedEmployee**(**stringfirstName**,** stringlastName**,**

stringssn**,** decimalweeklySalary**,** stringemployeeType**)**

**:** base**(**firstName**,** lastName**,** ssn**,** employeeType**)**

**{**

WeeklySalary=weeklySalary**;**

**}**

publicdecimalWeeklySalary

**{**

get **{** return\_weeklySalary**; }**

set

**{**

if **(**value>=0**)**

**{**

\_weeklySalary=value**;**

**}**

else

**{**

\_weeklySalary=0**;**

**}**

**}**

**}**

publicoverridestringToString**()**

**{**

stringoutput=base.ToString**();**

output+="\n\tWeekly Salary: "+PayCheck**()**.ToString**(**"C"**);**

returnoutput**;**

**}**

publicoverridedecimalPayCheck**()**

**{**

returnWeeklySalary**;**

**}**

**}**

sealedinternalclassHourlyEmployee **:** Employee

**{**

privatedecimal\_hourlyWage**;**

privatedecimal\_hoursWorked**;**

publicHourlyEmployee**(**stringfirstName**,** stringlastName**,** stringssn**,**

decimalhourlyWage**,** decimalhoursWorked**)**

**:** base**(**firstName**,** lastName**,** ssn**,** "Hourly Employee"**)**

**{**

HourlyWage=hourlyWage**;**

HoursWorked=hoursWorked**;**

**}**

publicdecimalHourlyWage

**{**

get **{** return\_hourlyWage**; }**

set

**{**

if **(**value>=0**)**

**{**

\_hourlyWage=value**;**

**}**

else

**{**

\_hourlyWage=0**;**

**}**

**}**

**}**

publicdecimalHoursWorked

**{**

get **{** return\_hoursWorked**; }**

set

**{**

if **(**value>=0**)**

**{**

\_hoursWorked=value**;**

**}**

else

**{**

\_hoursWorked=0**;**

**}**

**}**

**}**

publicdecimalEarnings

**{**

get

**{**

returnHoursWorked\*HourlyWage**;**

**}**

**}**

publicoverridedecimalPayCheck**()**

**{**

returnEarnings**;**

**}**

publicoverridestringToString**()**

**{**

stringoutput=base.ToString**();**

output+="\n\tHourly pay: "+HourlyWage.ToString**(**"C"**)** +

"\n\tHours worked: "+HoursWorked+

"\n\tPaycheck amount: "+PayCheck**()**.ToString**(**"C"**);**

returnoutput**;**

**}**

**}**

internalclassExecutiveEmployee **:** SalariedEmployee

**{**

privatedecimal\_bonus**;**

publicExecutiveEmployee**(**stringfirstName**,** stringlastName**,**

stringssn**,** decimalweeklySalary**,** decimalbonus**) :**

base**(**firstName**,** lastName**,** ssn**,** weeklySalary**,**

"Executive Employee"**)**

**{**

Bonus=bonus**;**

**}**

publicdecimalBonus

**{**

get **{** return\_bonus**; }**

set

**{**

if **(**value>0**)**

**{**

\_bonus=value**;**

**}**

**}**

**}**

publicnewstringToString**()**

**{**

stringoutput=base.ToString**();**

output+="\n\tAve. Weekly salary including Bonus: "+

PayCheck**()**.ToString**(**"C"**)** +

"\n\tBonus: "+Bonus.ToString**(**"C"**);**

returnoutput**;**

**}**

publicnewdecimalPayCheck**()**

**{**

decimalaverageWeeklyPay= **(**Bonus/52**)** +WeeklySalary**;**

returnaverageWeeklyPay**;**

**}**

**}**

classProgram

**{**

conststring\_EMPLOYEE\_FILES\_DIRECTORY\_FULL\_NAME=

@"..\debug\EmployeeFiles"**;**

staticvoidMain**(**string**[]** args**)**

**{**

// Create an instance of Employees specifying 7 employees.

Employeesemps=newEmployees**(**7**);**

SalariedEmployeese=newSalariedEmployee**(**"Jane"**,** "Doe"**,**

"444-55-6666"**,** 791.34m**);**

emps.AddEmployee**(**se**);**

HourlyEmployeehe=newHourlyEmployee**(**"Jim"**,** "Smith"**,**

"777-88-9999"**,** 12.38m**,** 40**);**

emps.AddEmployee**(**he**);**

ExecutiveEmployeeee=newExecutiveEmployee**(**"Laura"**,**

"Jones"**,** "333-44-5555"**,** 2813.71m**,** 25000**);**

emps.AddEmployee**(**ee**);**

he=newHourlyEmployee**(**"Kyle"**,** "McMasters"**,**

"111-55-3333"**,** 15.49m**,** 39**);**

emps.AddEmployee**(**he**);**

ee=newExecutiveEmployee**(**"Thomas"**,**

"Franklin"**,** "222-88-6666"**,** 3157.71m**,** 29000**);**

emps.AddEmployee**(**ee**);**

se=newSalariedEmployee**(**"Susan"**,** "Mason"**,**

"555-33-7777"**,** 842.25m**);**

emps.AddEmployee**(**se**);**

se=newSalariedEmployee**(**"Janet"**,** "Powell"**,**

"999-44-3333"**,** 678.12m**);**

emps.AddEmployee**(**se**);**

// Print basic info for all employees.

emps.PrintAllEmployees**();**

// Sort employees by last name, first name.

emps.SortEmployees**();**

// Print basic info for all sorted employees.

emps.PrintAllEmployees**();**

Console.Write**(**"\n\nPress <ENTER> to end: "**);**

Console.ReadLine**();**

**}**

**}**

**}**