Deakin University

Object Oriented Development

OnTrack Submission

# C# Essentials: Classes and Objects

*Submitted By:*
Peter Stacey
pstacey
2020/03/19 10:00

*Tutor:*
Dipto Pratyaksa

| Outcome | Weight |
|---|---|
| Evaluate Code | ♦♦♦◊◊ |
| Principles | ♦♦♦◊◊ |
| Build Programs | ♦♦♦◊◊ |
| Design | ♦◊◊◊◊ |
| Justify | ♦♦♦◊◊ |

The move into more object oriented aspects of development and the creation of classes, and separating code into separate files involves evaluating requirements, designing a solution and coding the end result into working programs that are tested.

March 19, 2020

Produced by Doubtfire

```cs
using System;

namespace Program_1
{
    class MobileProgram
    {
        static void Main(string[] args)
        {
            Mobile jimMobile = new Mobile("Monthly", "Samsung Galaxy S6",
            ↪  "07712223344");

            Console.WriteLine("Account Type: " + jimMobile.getAccType() +
                "\nMobile Number: " + jimMobile.getNumber() +
                "\nDevice: " + jimMobile.getDevice() +
                "\nBalance: " + jimMobile.getBalance());


            Console.WriteLine();

            jimMobile.setAccType("PAYG");
            jimMobile.setDevice("iPhone 6S");
            jimMobile.setNumber("07713334466");
            jimMobile.setBalance(15.50);

            Console.WriteLine("Account Type: " + jimMobile.getAccType() +
                "\nMobile Number: " + jimMobile.getNumber() +
                "\nDevice: " + jimMobile.getDevice() +
                "\nBalance: " + jimMobile.getBalance());

            Console.WriteLine();

            jimMobile.addCredit(10.0);
            jimMobile.makeCall(5);
            jimMobile.sendText(2);

            // Create additional mobile account and test
            Console.WriteLine("\nCreating new mobile account for Peter\n");

            Mobile peterMobile = new Mobile("Monthly", "Samsung Galaxy S9+",
            ↪  "0412324124");
            peterMobile.addCredit(50.00);
            peterMobile.makeCall(25);
            peterMobile.sendText(20);

            Console.ReadLine();
        }
    }
}
```

```
using System;

namespace Program_1
{
    /// <summary>
    /// The mobile class defines attributes and methods on mobile
    /// phone accounts
    /// </summary>
    class Mobile
    {
        // Instance variables
        private String accType, device, number;
        private double balance;

        // VARIABLES
        private const double CALL_COST = 0.245;
        private const double TEXT_COST = 0.078;


        /// <summary>
        /// Class constructor
        /// </summary>
        /// <param name="accType">The account type</param>
        /// <param name="device">The mobile phone make and model</param>
        /// <param name="number">The mobile phone number</param>
        public Mobile(String accType, String device, String number)
        {
            this.accType = accType;
            this.device = device;
            this.number = number;
            this.balance = 0.0;
        }

        /// <summary>
        /// Returns the account type
        /// </summary>
        /// <returns>
        /// The account type
        /// </returns>
        public String getAccType()
        {
            return this.accType;
        }

        /// <summary>
        /// Returns the device details
        /// </summary>
        /// <returns>
        /// The device make and model
        /// </returns>
        public String getDevice()
        {
            return this.device;
```

```
54          }
55
56          /// <summary>
57          /// Returns the mobile phone number
58          /// </summary>
59          /// <returns>
60          /// The the mobile phone number
61          /// </returns>
62          public String getNumber()
63          {
64              return this.number;
65          }
66
67          /// <summary>
68          /// Returns the account credit balance
69          /// </summary>
70          /// <returns>
71          /// The account credit balance in currency format
72          /// </returns>
73          public String getBalance()
74          {
75              return this.balance.ToString("C");
76          }
77
78          /// <summary>
79          /// Sets the account type
80          /// </summary>
81          /// <param name="accType">The new account type</param>
82          public void setAccType(String accType)
83          {
84              this.accType = accType;
85          }
86
87          /// <summary>
88          /// Sets the device details
89          /// </summary>
90          /// <param name="device">The device make and model</param>
91          public void setDevice(String device)
92          {
93              this.device = device;
94          }
95
96          /// <summary>
97          /// Sets the mobile phone number
98          /// </summary>
99          /// <param name="number">The new mobile phone number</param>
100         public void setNumber(String number)
101         {
102             this.number = number;
103         }
104
105         /// <summary>
106         /// Sets the account type
```

```
107         /// </summary>
108         /// <param name="balance">The new balance to set</param>
109         public void setBalance(double balance)
110         {
111             this.balance = balance;
112         }
113 
114         /// <summary>
115         /// Adds credit to the account balance
116         /// </summary>
117         /// <param name="amount">The amount to credit the account</param>
118         public void addCredit(double amount)
119         {
120             this.balance += amount;
121             Console.WriteLine("Credit added successfully. New balance " +
                ↪  getBalance());
122         }
123 
124         /// <summary>
125         /// Calculates the cost of a call by minutes talking and updates
126         /// the balance
127         /// </summary>
128         /// <param name="minutes">The time of the call(s) in minutes</param>
129         public void makeCall(int minutes)
130         {
131             double cost = minutes * CALL_COST;
132             this.balance -= cost;
133             Console.WriteLine("Call made. New balanace " + getBalance());
134         }
135 
136         /// <summary>
137         /// Calculates the cost of text sent by the number of texts and
138         /// updates the balance
139         /// </summary>
140         /// <param name="numTexts">The number of texts sent</param>
141         public void sendText(int numTexts)
142         {
143             double cost = numTexts * TEXT_COST;
144             this.balance -= cost;
145             Console.WriteLine("Text Sent. New balance " + getBalance());
146         }
147     }
148 }
```

```cs
using System;

namespace Program_2
{
    class EmployeeProgram
    {
        static void Main(string[] args)
        {
            // Create two employees with different salaries
            Employee andrew = new Employee("Andrew Cain", 180000);
            Employee jane = new Employee("Jane Doe", 45000);

            // Test getting the name and salary
            Console.WriteLine("Employee Name: " + andrew.getName() +
                ", Salary: " + andrew.getSalary());
            Console.WriteLine("Employee Name: " + jane.getName() +
                ", Salary: " + jane.getSalary());

            // Test increasing the salary
            andrew.raiseSalary(5.0);  // expect $189000
            jane.raiseSalary(15.0);   // expect $51750

            Console.WriteLine();

            // Create additional employee in lowest tax bracket
            Employee trev = new Employee("Trev", 12300);
            Console.WriteLine("Employee Name: " + trev.getName() +
                ", Salary: " + trev.getSalary());

            // Test tax calculates correctly
            Console.WriteLine("Employee Name: " + andrew.getName() +
                ", Tax Burden: " + andrew.Tax()); // expect $58146
            Console.WriteLine("Employee Name: " + jane.getName() +
                ", Tax Burden: " + jane.Tax()); // expect $8365.75
            Console.WriteLine("Employee Name: " + trev.getName() +
                ", Tax Burden: " + trev.Tax()); // expect Nil tax

        }
    }
}
```

```cs
using System;

namespace Program_2
{
    /// <summary>
    /// Class for employee details and salary
    /// </summary>
    class Employee
    {
        // Instance variables
        private String name;
        private double salary;

        /// <summary>
        /// The class constructor
        /// </summary>
        /// <param name="employeeName">The name of the employee</param>
        /// <param name="currentSalary">The current salary</param>
        public Employee(string employeeName, double currentSalary)
        {
            this.name = employeeName;
            this.salary = currentSalary;
        }

        /// <summary>
        /// Returns the name of the employee
        /// </summary>
        /// <returns>
        /// The name of the employee
        /// </returns>
        public String getName()
        {
            return this.name;
        }

        /// <summary>
        /// Returns the current salary of the employee
        /// </summary>
        /// <returns>
        /// The current salary of the employee as a string
        /// </returns>
        public String getSalary()
        {
            return this.salary.ToString("C");
        }

        /// <summary>
        /// Raises the current salary by a percentage
        /// </summary>
        /// <param name="percentRaise">The percent amount to add to the
        ↪  salary</param>
        public void raiseSalary(double percentRaise)
        {
```

```
53              this.salary = this.salary * (1.0 + (percentRaise / 100));
54              Console.WriteLine("Current salary for " + getName() + " now " +
                ↪  getSalary());
55          }
56
57          /// <summary>
58          /// Calculates the amount of tax deducted annually from the salary
59          /// </summary>
60          /// <returns>
61          /// The annual tax burden as a double
62          /// </returns>
63          public String Tax()
64          {
65              if (this.salary >= 180000)
66              {
67                  double tax = 54096 + (0.45 * (this.salary - 180000));
68                  return tax.ToString("C");
69              }
70              else if (this.salary > 90000)
71              {
72                  double tax = 20797 + (0.37 * (this.salary - 90000));
73                  return tax.ToString("C");
74              }
75              else if (this.salary > 37000)
76              {
77                  double tax = 3572 + (0.325 * (this.salary - 37000));
78                  return tax.ToString("C");
79              }
80              else if (this.salary > 18200)
81              {
82                  double tax = 0.18 * (this.salary - 18200);
83                  return tax.ToString("C");
84              }
85              else
86              {
87                  return "Nil";
88              }
89          }
90      }
91  }
```

```
using System;

namespace Program_3
{
    class CarProgram
    {
        static void Main(string[] args)
        {
            // Create a myCar object
            Car myCar = new Car(14.5, 65.5);

            // Test setting total miles, getting total miles and fuel
            myCar.setTotalMiles(100);
            Console.WriteLine("Current mileage: " + myCar.getTotalMiles()
                + ", Current fuel: " + myCar.getFuel());

            // Test adding fuel and printing the fuel cost
            myCar.addFuel(22);
            Console.WriteLine("Current fuel cost per litre: " +
            ↪  myCar.printFuelCost());

            // Test driving
            myCar.drive(60);
            Console.WriteLine("Current mileage: " + myCar.getTotalMiles()
                + ", Current fuel: " + myCar.getFuel());

            // Test driving again to check mileage accumulates correctly
            myCar.drive(30);
            Console.WriteLine("Current mileage: " + myCar.getTotalMiles()
                + ", Current fuel: " + myCar.getFuel());

            Console.ReadLine();
        }
    }
}
```

```
using System;

namespace Program_3
{
    /// <summary>
    /// Defines properties and methods to track car mileage, fuel and cost
    /// </summary>
    class Car
    {
        // Instance variables
        private double fuelEfficiency;
        private double fuelLevel;
        private int mileage;

        // VARIABLES
        double FUEL_COST = 1.385;
        double GALLONS_TO_LITRES = 4.546;

        /// <summary>
        /// Class constructor
        /// </summary>
        public Car(double efficiency, double fuel)
        {
            this.fuelEfficiency = efficiency;
            this.fuelLevel = fuel;
            this.mileage = 0;
        }

        /// <summary>
        /// Returns the current fuel in litres
        /// </summary>
        /// <returns>
        /// The current fuel in litres
        /// </returns>
        public double getFuel()
        {
            return this.fuelLevel;
        }

        /// <summary>
        /// Returns the total mileage
        /// </summary>
        /// <returns>
        /// The current mileage of the car
        /// </returns>
        public int getTotalMiles()
        {
            return this.mileage;
        }

        /// <summary>
        /// Sets the total mileage
        /// </summary>
```

```
54          /// <param name="miles">The total miles to set</param>
55          public void setTotalMiles(int miles)
56          {
57              this.mileage = miles;
58          }
59
60          /// <summary>
61          /// Returns the cost of fuel in currency format
62          /// </summary>
63          /// <returns>
64          /// The current cost of fuel
65          /// </returns>
66          public String printFuelCost()
67          {
68              return this.FUEL_COST.ToString("C");
69          }
70
71          /// <summary>
72          /// Returns the total cost of fuel use
73          /// </summary>
74          /// <returns>
75          /// The total cost of using an amount of fuel
76          /// </returns>
77          /// <param name="fuelLitres">The litres of fuel used</param>
78          public double calcCost(double fuelLitres)
79          {
80              return fuelLitres * this.FUEL_COST;
81          }
82
83          /// <summary>
84          /// Adds fuel to the fuel tank
85          /// </summary>
86          /// <param name="fuelLitres">Volume of fuel in litres</param>
87          public void addFuel(double fuelLitres)
88          {
89              this.fuelLevel += fuelLitres;
90              double fillCost = calcCost(fuelLitres);
91              Console.WriteLine("Cost of fill: "
92                  + calcCost(fuelLitres).ToString("C"));
93          }
94
95          /// <summary>
96          /// Converts fuel volume from gallons to litres
97          /// </summary>
98          /// <returns>
99          /// The volume of fuel in litres
100         /// </returns>
101         /// <param name="gallons">The gallons of fuel to convert</param>
102         public double convertToLitres(double gallons)
103         {
104             return gallons * this.GALLONS_TO_LITRES;
105         }
106
```

```
107         /// <summary>
108         /// Calculates and outputs the cost of a trip and updates car
109         /// properties
110         /// </summary>
111         /// <param name="milesTravelled">The total miles travelled</param>
112         public void drive(int milesTravelled)
113         {
114             this.mileage += milesTravelled; // accumulate mileage
115             double gallonsUsed = milesTravelled / this.fuelEfficiency;
116             double litresUsed = convertToLitres(gallonsUsed);
117             this.fuelLevel -= litresUsed; // remove fuel from the tank
118             double tripCost = calcCost(litresUsed);
119             Console.WriteLine("Total cost of travelling "
120                 + milesTravelled + " miles = "
121                 + tripCost.ToString("C"));
122         }
123     }
124 }
```