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

/*  TITLE:      RunningCostsCalculator
 *  AUTHOR:     Paul McKillop
 *  DATE:       October 2022
 *  PURPOSE:    Calculate the costs other than the Loan costs
 */

namespace McKillopMotoring
{
    public class RunningCostsCalculator
    {

        #region Insurance
        // - Insurance Costs

        // --- Annual costs
        public static double InsuranceCostAnnual(RunningCost myRunningCost)
        {
            //-- return value
            double cost = 0;
            string period = myRunningCost.InsurancePeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Insurance);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Insurance * 12);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Insurance * 52);
                    break;
                //-- switch must have a default return value to avoid endless loop or unreachable code
                default:
                    cost = 0;
                    break;
            }

            // --now return the value from the method based on the switch
            return Math.Round(cost, 2);
        }

        // --- Monthly costs
        public static double InsuranceCostMonthly(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.InsurancePeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Insurance / 12);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Insurance);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Insurance * 52 / 12);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }
        // --- Weekly costs
        public static double InsuranceCostWeekly(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.InsurancePeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Insurance / 52);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Insurance * 12 / 52);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Insurance);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }
        #endregion

        //-- Fuel
        //-- Annual
        #region Fuel costs

        public static double FuelCostAnnual(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.FuelPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Fuel);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Fuel * 12);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Fuel * 52);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }

        //-- Monthly
        public static double FuelCostMonthly(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.FuelPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Fuel / 12);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Fuel);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Fuel * 52 / 
                  12);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }

        //-- Weekly
        public static double FuelCostWeekly(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.FuelPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Fuel / 52);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Fuel * 12 / 52);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Fuel);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }
        #endregion


        //-- Servicing
        #region Servicing costs
        public static double ServicingCostAnnual(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.ServicingPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Servicing);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Servicing * 12);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Servicing * 
                  52);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }

        public static double ServicingCostMonthly(RunningCost 
          myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.ServicingPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(myRunningCost.Servicing / 
                  12);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(myRunningCost.Servicing);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(myRunningCost.Servicing * 
                  52 / 12);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(cost, 2);
        }

        public static double ServicingCostWeekly(RunningCost 
          myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.ServicingPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(value: 
                  myRunningCost.Servicing / 52);
```

```
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(value: myRunningCost.Servicing * 12 / 52);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(value: myRunningCost.Servicing);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(value: cost, digits: 2);
        }
        #endregion

        //-- RoadTax
        #region Road Tax Costs
        public static double RoadTaxAnnual(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.RoadTaxPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax * 12);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax * 52);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(value: cost, digits: 2);
        }

        public static double RoadTaxMonthly(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.RoadTaxPeriod;
            {
                case "Annual":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax / 12);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax * 52 / 12);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(value: cost, digits: 2);
        }

        public static double RoadTaxWeekly(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = myRunningCost.RoadTaxPeriod;

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax / 52);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax * 12 / 52);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(value: myRunningCost.RoadTax);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(value: cost, digits: 2);
        }
        #endregion

        //-- Total Weekly Running costs
        public static double TotalWeeklyRunningCost(RunningCost
            double totalWeeklyCost = 0;
            totalWeeklyCost += InsuranceCostWeekly(myRunningCost);
            totalWeeklyCost += FuelCostWeekly(myRunningCost);
            totalWeeklyCost += ServicingCostWeekly(myRunningCost);
            totalWeeklyCost += RoadTaxWeekly(myRunningCost);

            return totalWeeklyCost;
        }



        #region Method Template
        //-- Template
        public static double CalculatorTemplate(RunningCost myRunningCost)
        {
            double cost = 0;
            string period = "Z";

            switch (period)
            {
                case "Annual":
                    cost = Convert.ToDouble(value: 0);
                    break;
                case "Monthly":
                    cost = Convert.ToDouble(value: 0);
                    break;
                case "Weekly":
                    cost = Convert.ToDouble(value: 0);
                    break;
                default:
                    cost = 0;
                    break;

            }

            return Math.Round(value: cost, digits: 2);
        }
        #endregion

    }
}
```