ASU W. P. CAREY SCHOOL of BUSINESS

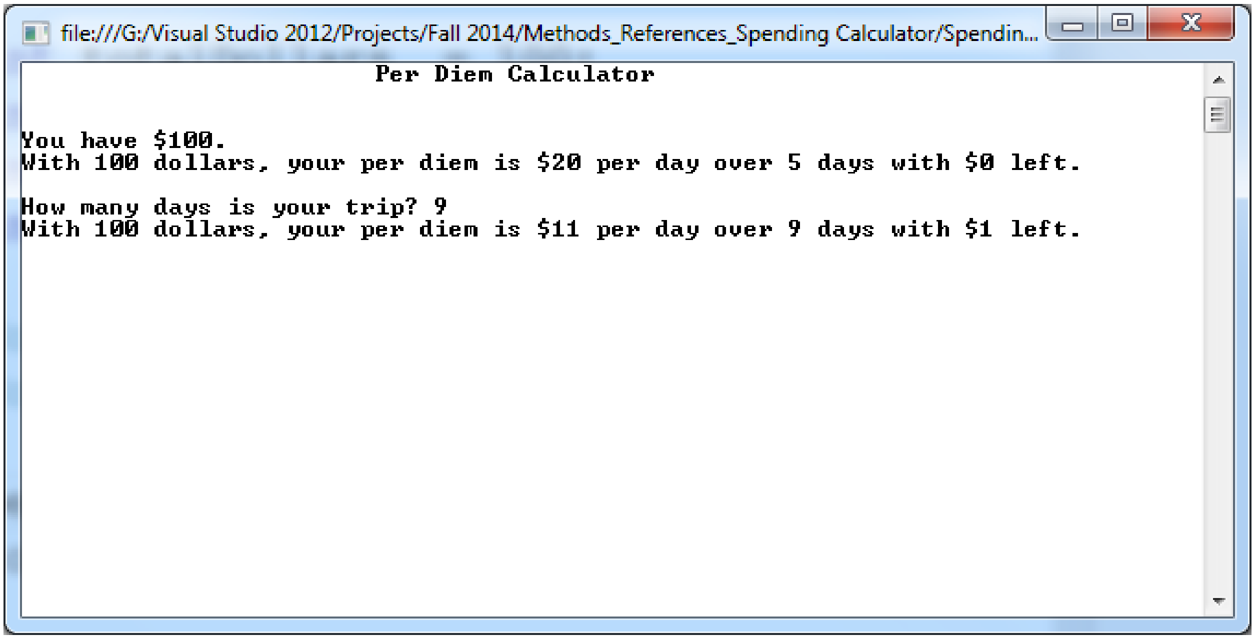
ARIZONA STATE UNIVERSITY

CIS 345 – Business Information Systems Development II – Fall 2014

In-class Exercise 3: Methods (Passing by Reference)

Due on Blackboard: Today, Wednesday, September 17 by 10:00 PM

Objective: This brief exercise is designed for you to consider an application of passing variables by value vs passing by reference. It also uses optional parameters.


file:///G:/Visual Studio 2012/Projects/Fall 2014/Methods_References_Spending Calculator/Spendin...

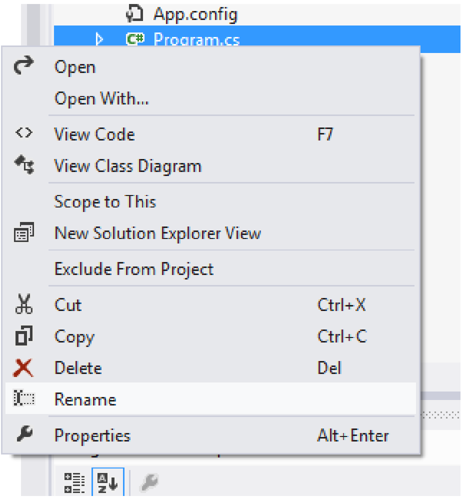
```
                        Per Diem Calculator


You have $100.
With 100 dollars, your per diem is $20 per day over 5 days with $0 left.

How many days is your trip? 9
With 100 dollars, your per diem is $11 per day over 9 days with $1 left.
```

1. Create a project called SpendingCalculatorApp.

2. Rename your class name from Program.cs to SpendingCalculator.cs by right-clicking on the filename in Solution Explorer and clicking rename. Accept its recommendation to change the name of the class throughout the project code.



Create or complete all the following methods:

Note: All methods in this project will be static.

Make sure you put the method inside the class block. Methods must always be within the class block but they cannot be placed inside another method i.e. methods cannot be nested.

All methods will be in the structure:

```
private static ReturnValueType MethodName (ParameterType ParameterName )
 {
    // Insert in the body of the method, statements as described in the method descriptions.
  }
```

Do not start working on the Main method first. Since Main calls other methods, we must write those methods first.

1. *Method name*: CalculatePerDiem
Purpose: This method calculates how many dollars one can spend per day for a given total number of dollars and a given number of days. It returns the per diem value as its formal return value. It stores the dollars left in the parameter passed to it by reference.
Parameters: 3 integers, with the following characteristics:

dollars, required parameter
change, required parameter, passed by reference
days, optional parameter, default value of 5

Return value type: int

In the body of the method:

- Divide dollars by days using the remainder operator. Assign the resulting value to change.
- Divide dollars by days (integer division). Return the resulting value using the return keyword.

2. Complete the Main method:

Even though the main method runs first, you should write it last, so the methods it calls are already written.

In the body of the Main method:

- Declare totalDollars, leftoverChange, perDiem, numberOfDays all as integers. Initialize total dollars to 100 and all other variables to zero.
- Put in WriteLine statements to make the Console look similar to the output i.e. welcome statement, message saying they have a hundred dollars, etc.

- Call CalculatePerDiem. Pass to it only the variables for total dollars and change. The variable for change will need to be passed by reference. The number of days parameter is optional – do not pass it. Store the return value in perDiem.
- Tell the user how much the per diem is for 5 days and how much money will be left over.

- Ask the user for the number of days. Store it in the appropriate variable you have declared.
- Call PerDiem again. This time, pass it all the appropriate arguments including the number of days that has just been specified by the user.
- Tell the user how much the per diem is for the specified number of days and how much money will be left over.

Additional Learning Exercises

Investigating Scope Further

You will find that even though the CalculatePerDiem method did not have access to the perDiem variable that is local to the main method, it was able to put a value in that variable because of the way references work.

Go and verify the scope of perDiem. In the body of the CalculatePerDiem method, use a WriteLine to output perDiem. If it doesn't work, comment it out. If it does work, put a comment in explaining why it worked. *Convince yourself of the way scope works - that perDiem is truly inaccessible in that method. Then, try various ways to get access to variables you don't have access to, by passing variables by reference.*

Make a change to the Method

After your program is working, to get more practice, change the behavior of the method, so it accepts both the change as well as the per diem variables by reference. In this case, you will change the return type to either 1) void, and not return anything or 2) keep it as an int and return a zero.

Alternatively, create an overloaded method that accepts the two variables by references.

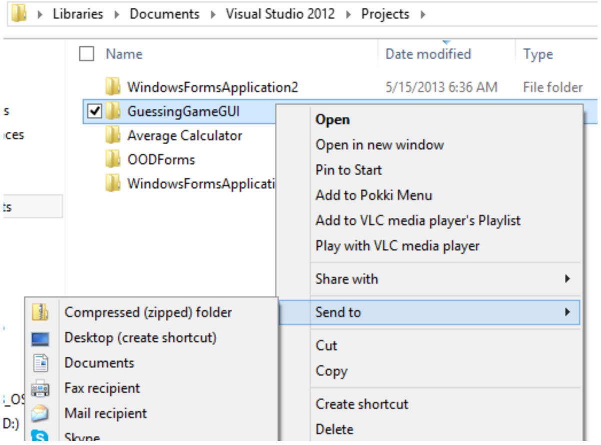
Investigate Bi-directionality

Do references work in both directions? You have implemented a method which accepts a reference and changes its value from within the method. Will it work in the other direction? If you change a value outside the method, will the change be reflected in the method? See if you can devise a test to see if it works this way or not.

Submission Instructions

Submission should be made using a zip file that contains all of the Visual Studio C# project files. You will need to ***zip the entire project folder*** along with the .sln and .suo files. The folder will automatically contain the class source files as well as the executable file that is generated in \ProjectName\bin\Debug folder. Upload file to the Blackboard assignment drop box.

Zip the entire top-level folder by right-clicking the folder and selecting Send to | Compressed (zipped) folder.



Using built-in windows zip tools: http://windows.microsoft.com/en-us/windows/compress-uncompress-files-zip-files

Make sure you check the following. Your grade is dependent on all these criteria being met.

- You have included your name as a comment within your class.
    - e.g. " // Inclass 3, Jane C. Smith, CIS 345, Tuesday 9:00 AM"
- Class file is called SpendingCalculator.cs (rename from Program.cs).
- Your Visual Studio project is called SpendingCalculatorApp.
- Zip filename is: **FirstNameLastName_Inclass3.zip**

Verify your zip file before you submit

- Check for actual class files being present in the folder before you zip it.
- Check your zip file size after zipping – if it is 1K, it likely contains only a shortcut.
- Uncompress your zip file before submitting and verify that files are present.
- Download your zip file after submitting, uncompress, and again verify that your files are present.