# Lab: Debugging and Troubleshooting Code

Problems for exercises and homework for the [“Programming Fundamentals Extended” course @ SoftUni](https://softuni.bg/courses/programming-fundamentals).

You can check your solutions here: <https://judge.softuni.bg/Contests/304/Methods-and-Debugging-Lab>.

## Multiply Evens by Odds

Create a program that reads an **integer number** and **multiplies the sum of all its even digits** by **the sum of all its odd digits**:

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 12345 | 54 | 12345 has **2 even digits** - 2 and 4. Even digits has **sum of 6**.  Also it has **3 odd digits** - 1, 3 and 5. Odd digits has **sum of 9**.  **Multiply 6 by 9** and you get **54**. |
| -12345 | 54 |  |

### Hints

1. Create a method with a **name describing its purpose** (like GetMultipleOfEvensAndOdds). The method should have a **single integer parameter** and an **integer return value**. Also the method will call two other methods:



1. Create two other methods each of which will sum either even or odd digits
2. Implement the logic for summing odd digits:



1. Do the same for the method that will sum even digits
2. As you test your solution you may notice that it doesn't work for negative numbers. Following the program execution line by line, find and fix the bug (**hint: you can use Math.Abs()**)

## Debugging Exercise: Holidays Between Two Dates

You are assigned to **find and fix the bugs** in an existing piece of code, using the Visual Studio **debugger**. You should trace the program execution to find the lines of code that produce incorrect or unexpected results.

You are given a program (existing **source code**) that aims to **count the non-working days between two dates** given in format day.month.year (e.g. between **1.05.2015** and **15.05.2015** there are **5** non-working days – Saturday and Sunday).

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 01.05.2016  15.05.2016 | 5 | There are **5** non-working days (Saturday / Sunday) in this period: 1-May-2016, 7-May-2016, 8-May-2016, 14-May-2016, 15-May-2016 |
| 01.05.2016  02.05.2016 | 1 | Only **1** non-working day in the specified period: 1.05.2016 (Sunday) |
| 15.05.2020  10.05.2020 | 0 | The second date is before the first. No dates in the range. |
| 22.02.2020  01.03.2020 | 4 | Two Saturdays and Sundays:   * 22.02.2020 and 23.02.2020 * 29.02.2020 and 1.03.2020 |

You can **find the broken code** in the judge system: [Broken Code for Refactoring](http://svn.softuni.org/admin/svn/soft-tech/Jan-2017/Programming-Fundamentals-Extended-Jan-2017/05.%20Programming-Fundamentals-Debugging-and-Troubleshooting-Code/05.%20Programming-Fundamentals-Debugging-and-Troubleshooting-Code-Lab-Broken-Solutions.zip). It looks as follows:

|  |
| --- |
| HolidaysBetweenTwoDates.cs |
| using System;  using System.Globalization;  class HolidaysBetweenTwoDates  {  static void Main()  {  var startDate = DateTime.ParseExact(Console.ReadLine(),  "dd.m.yyyy", CultureInfo.InvariantCulture);  var endDate = DateTime.ParseExact(Console.ReadLine(),  "dd.m.yyyy", CultureInfo.InvariantCulture);  var holidaysCount = 0;  for (var date = startDate; date <= endDate; date.AddDays(1))  if (date.DayOfWeek == DayOfWeek.Saturday &&  date.DayOfWeek == DayOfWeek.Sunday) holidaysCount++;  Console.WriteLine(holidaysCount);  }  } |

### Hints

There are **4** **mistakes** in the code. You’ve got to **use the debugger** to find them and fix them. After you do that, submit your **fixed code in the judge contest**: [https://judge.softuni.bg/Contests/Practice/Index/419#1](https://judge.softuni.bg/Contests/Practice/Index/419%231).

## Debugging Exercise: Price Change Alert

You are assigned to **rework a given piece of code** which is working **without bugs** but is **not properly formatted**.

The given program **tracks stock prices** and **gives updates** about the **significance in each price change**. Based on the significance, there are **four kind of changes**: no change at all (price is equal to the previous), minor (difference is below the significance threshold), price up and price down.

You can **find the broken code** here: [Broken Code for Refactoring](http://svn.softuni.org/admin/svn/soft-tech/Jan-2017/Programming-Fundamentals-Extended-Jan-2017/05.%20Programming-Fundamentals-Debugging-and-Troubleshooting-Code/05.%20Programming-Fundamentals-Debugging-and-Troubleshooting-Code-Lab-Broken-Solutions.zip).

### Input

* On the first line you are given **N** - the number of prices
* On the second line you are given the significance threshold
* On the next N lines, you are given prices

### Output

* Don’t print anything for the first price
* If there is **no difference** from the previous price the output message is: "NO CHANGE: {current price}"
* In case of **minor change**: "MINOR CHANGE: {last price} to {current price} ({difference}%)"
* In case of **major change**: "PRICE UP: {last price} to {current price} ({difference}%)" or "PRICE DOWN: {last price} to {current price} ({difference}%)"

The percentage should be rounded to the second digit after the decimal point.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  0.1  10  11  12 | PRICE UP: 10 to 11 (10.00%)  MINOR CHANGE: 11 to 12 (9.09%) |
| 3  0.1  10  10  12 | NO CHANGE: 10  PRICE UP: 10 to 12 (20.00%) |

### Hints

1. Download the source code and get familiar with it
2. Deal with poor code formatting - Remove unnecessary blank lines, indent the code properly
3. Fix method parameters naming
4. Give methods a proper name