# Software testing documentation

## Test case 1

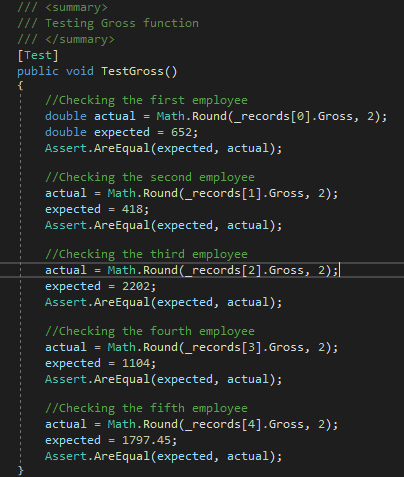
Table : Test case 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 1 | Test Import – The purpose of this test is to see if CsvImporter Class can import the file and read the content accurately. | | | 1 |
| **Test type/technique** | | | | |
| File upload | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| Employee-payroll-data.csv file | | 1.\_records are not null  2. \_records contain 5 object or employee data  3. \_records are not empty | 1.\_records are not null  2.\_records contains 5 objects/employee data  3.\_records are not empty | Pass |
| **Steps** | | | | |
| **1** | Import the record via CsvImporter.ImportPayRecord method into the \_records list | | | |
| **2** | Setup Asset.IsNotNull | | | |
| **3** | Setup Assert.IsNotEmpty | | | |
| **4** | Set Assert.AreEqual | | | |
| **5** | Run the test and check results | | | |

## 

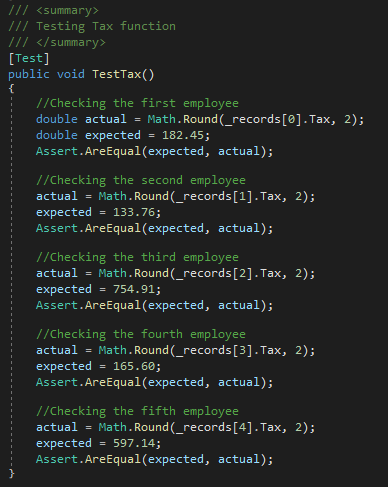
**Test Case 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 2 | TestGross() – Purpose of this test is to check the gross amount for all 5 employees from the csv file | | | 2 |
| **Test type/technique** | | | | |
| Using Calculation | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| Data that stored in the \_record list | | \_records[0].Gross=652  \_records[1].Gross=418  \_records[2].Gross=2202  \_records[3].Gross=1104  \_records[4].Gross=1797.45 | \_records[0].Gross=652  \_records[1].Gross=418  \_records[2].Gross=2202  \_records[3].Gross=1104  \_records[4].Gross=1797.45 | Pass |
| **Steps** | | | | |
| **1** | Setup actual results from \_records[0].Gross | | | |
| **2** | Setup expected result from \_records[0].Gross | | | |
| **3** | Repeat steps 1 & 2 for other employees/indexes | | | |
| **4** | Run the test | | | |
| **5** | Check for the result and compare the two values | | | |



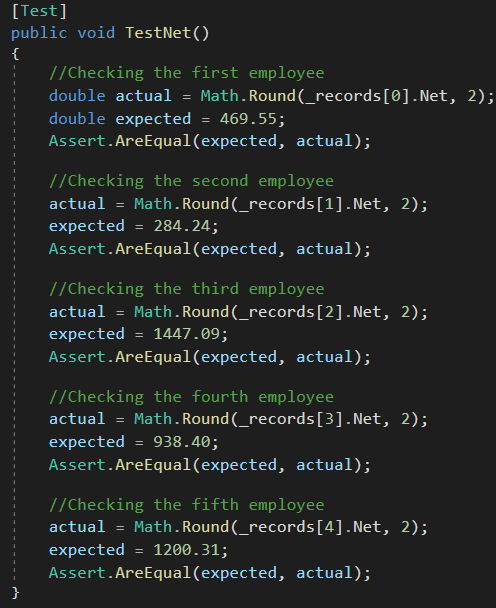
**Test Case 3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 3 | TestTax() – This test is to check the Tax amount and compare the values | | | 3 |
| **Test type/technique** | | | | |
| Using Calculation | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| Data that stored in the \_record list | | \_records[0].Tax=182.45  \_records[1].Tax=133.76  \_records[2].Tax=754.91  \_records[3].Tax=165.60  \_records[4].Tax=597.14 | \_records[0].Tax=182.45  \_records[1].Tax=133.76  \_records[2].Tax=754.91  \_records[3].Tax=165.60  \_records[4].Tax=597.14 | Pass |
| **Steps** | | | | |
| **1** | Setup actual results from \_records[0].Tax | | | |
| **2** | Setup expected result from \_records[0].Tax | | | |
| **3** | Repeat steps 1 & 2 for other employees/indexes | | | |
| **4** | Run the test | | | |
| **5** | Check for the result and compare the two values | | | |



**Test Case 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 4 | TestNet() – Purpose of this test is to check the Net amount for all 5 employees from the csv file | | | 4 |
| **Test type/technique** | | | | |
| Using Calculation | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| Data that stored in the \_record list | | \_records[0].Net=469.55  \_records[1].Net=284.24  \_records[2].Net=1447.09  \_records[3].Net=938.40  \_records[4].Net=1200.31 | \_records[0].Net=469.55  \_records[1].Net=284.24  \_records[2].Net=1447.09  \_records[3].Net=938.40  \_records[4].Net=1200.31 | Pass |
| **Steps** | | | | |
| **1** | Setup actual results from \_records[0].Net | | | |
| **2** | Setup expected result from \_records[1].Net | | | |
| **3** | Repeat steps 1 & 2 for other employees/indexes | | | |
| **4** | Run the test | | | |
| **5** | Check for the result and compare the two values | | | |



**Test Case 5**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 5 | TestExport() – Purpose of this test is to test the Export method PayRecordWriter and if successfully writes the calculated result to a csv file in the Export Folder | | | 5 |
| **Test type/technique** | | | | |
| CsvHelper Write method | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| Data that stored in the \_record list | | Writes data from \_records to csv file and store them in Export folder | \_records has been exported into csv file and stored in Export folder | Pass |
| **Steps** | | | | |
| **1** | Define the path and file name to the export folder | | | |
| **2** | Define the path and file name into a bool variable to check if file exist | | | |
| **3** | The bool value of the File.Exists is then compared to Assert.IsTrue | | | |
| **4** | Run the test and compare expected to actual | | | |

**Test Data**

Table 3: Test Data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Hours | Rate | Visa | Ytd | Gross | Total YTD + gross | Total gross | Tax | Net |
| 1 | 2 | 25 |  |  | 50 | N**/**A | **652.00** | **182.45** | **469.55** |
| 1 | 3 | 25 |  |  | 75 |
| 1 | 3 | 25 |  |  | 75 |
| 1 | 4 | 25 |  |  | 100 |
| 1 | 5 | 32 |  |  | 160 |
| 1 | 6 | 32 |  |  | 192 |
| 2 | 2 | 25 | 417 | 47520 | 50 | **47938** | **418.00** | **133.76** | **284.24** |
| 2 | 2 | 25 | 417 | 47520 | 50 |
| 2 | 2 | 25 | 417 | 47520 | 50 |
| 2 | 2 | 25 | 417 | 47520 | 50 |
| 2 | 2 | 25 | 417 | 47520 | 50 |
| 2 | 2 | 28 | 417 | 47520 | 56 |
| 2 | 2 | 28 | 417 | 47520 | 56 |
| 2 | 2 | 28 | 417 | 47520 | 56 |
| 3 | 8 | 36 |  |  | 288 |  | **2202.00** | **754.91** | **1447.09** |
| 3 | 8 | 36 |  |  | 288 |
| 3 | 8 | 36 |  |  | 288 |
| 3 | 8 | 36 |  |  | 288 |
| 3 | 8 | 37.5 |  |  | 300 |
| 3 | 8 | 37.5 |  |  | 300 |
| 3 | 6 | 37.5 |  |  | 225 |
| 3 | 6 | 37.5 |  |  | 225 |
| 4 | 5 | 34.5 | 462 | 23000 | 172.5 | 24104.00 | **1104.00** | **165.60** | **938.40** |
| 4 | 5 | 34.5 | 462 | 23000 | 172.5 |
| 4 | 5 | 34.5 | 462 | 23000 | 172.5 |
| 4 | 5 | 34.5 | 462 | 23000 | 172.5 |
| 4 | 5 | 34.5 | 462 | 23000 | 172.5 |
| 4 | 5 | 34.5 | 462 | 23000 | 172.5 |
| 4 | 2 | 34.5 | 462 | 23000 | 69 |
| 5 | 7 | 42.5 |  |  | 297.5 |  | **1797.45** | **597.14** | **1200.31** |
| 5 | 6.5 | 42.5 |  |  | 276.25 |
| 5 | 7 | 42.5 |  |  | 297.5 |
| 5 | 7 | 42.5 |  |  | 297.5 |
| 5 | 7 | 42.5 |  |  | 297.5 |
| 5 | 3 | 55.2 |  |  | 165.6 |
| 5 | 3 | 55.2 |  |  | 165.6 |

## Debugging processes

[Describe how you decided on the action to take when debugging your code.

Explain the key principles and concepts of the debugging tools and how you applied these.]

When debugging, we look for known common errors first such as missing “;” or “}” or “{“ and general syntax errors.

These types of errors are normally revealed within the IDE that you are using such as Visual Studio.

There options that allow you to run at the last time the code was successfully run, you can return to that point.

Alternatively, you should have a test and debug plan. Know what each part of the code should do and what each variable or object is suppose to have.

Then you can narrow down and find the root cause.

The IDE such as Visual Studios allow programmers/developers to implement breakpoints for them to follow their code as it gets executed step by step.

You can separate your codes into other classes and invoke them as needed therefore isolating it to an area of code. If you have just added something, you can also comment out the last code you added and check to see if your program can still run.

The whole point of debugging is to ensure your programming runs smoothly without crashing. With debugging it reports an error condition immediately. This allows earlier detection of an error and makes the process of software development stress-free and unproblematic.