# Software testing documentation

## Test case 1

Table 1: Test case 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 1 | Logic test | | | 1 |
| **Test type/technique** | | | | |
| File upload | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| See **Table 3: Test Data** for complete data and expected results | | Values will calculate as listed | As listed | Pass |
| **Steps** | | | | |
| **1** | Wrote [SetUp] method to set filePath and populate List with records | | | |
| **2** | [Test] methods to test calculations for Gross, Net, Tax: | | | |
| **3** | Double variables expected and actual, reading from List and using hard coded values as listed, for each employee | | | |
| **4** | Build solution | | | |
| **5** | Debug all tests | | | |
| **6** |  | | | |

## Test case 2

Table 2: Test case 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Name and description | | | Priority |
| 2 | Reusable component test | | | 2 |
| **Test type/technique** | | | | |
| File upload | | | | |
| **Test data** | | **Expected** | **Actual** | **Pass/Fail** |
| Data contained in *employee-payroll-data.csv* file | | Values will display to console and write to csv file. | Displayed to console and wrote to csv | Pass |
| **Steps** | | | | |
| **1** | Wrote [SetUp] method to set filePath and populate List with records | | | |
| **2** | [Test] method TestImport check if List IsNotNull, IsNotEmpty, AreEqual to five records | | | |
| **3** | [Test] method TestExport check if file exists in export folder (bool) | | | |
| **4** | Build solution | | | |
| **5** | Debug all tests | | | |
| **6** |  | | | |

### 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 | N/A | **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 | N/A | **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.

* Place a break point after [SetUp] to see number of records imported to the List
* Tests passed as expected, no need for further debugging

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

* To check if code is working properly.
* To check if something is wrong, e.g. file path

