**Test Function Document**

|  |  |
| --- | --- |
| Project Name: Assignment 1 Testing | |
| Automation Title: test\_display\_csv\_as\_table | Version: 1.0 |
| Testing Phase: Phase 1 | Date of Test: 16 November 2023 |
| Module Name: products.py | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Function Title: display\_csv\_as\_table | | | | Test Designed by: William Brunnsberg | | | |
| Test Priority (Low/Medium/High): Low | | | | Test Designed Date: 16 November 2023 | | | |
| Description: The objective of the display\_csv\_as\_table function is to print every row in a csv-file in the terminal. It identifies the first row as the header. | | | | Test Executed by: William Brunnsberg | | | |
| Test Execution date: 16 November 2023 | | | |
|  | | | | | | | |
| Pre-conditions: None | | | | | | | |
| Dependencies: None | | | | | | | |
|  | | | | | | | |
| S. No | Equivalence Class | Test case data | Expected Results | | Actual Results | Status (Pass/Fail) | Notes |
| 1 | IEC1 | 1 | Function raises TypeError | | Function raises StopIteration error | Fail | Test name: test\_int\_input()  Int input |
| 2 | IEC2 | 0.5 | Function raises TypeError | | Function raises TypeError | Pass | Test name: test\_float\_input()  Float input |
| 3 | IEC3 | ["copy\_products.csv", "copy\_products.csv"] | Function raises TypeError | | Function raises TypeError | Pass | Test name: test\_list\_input()  List input.  Uses a fixture to create a temporary copy of products.csv |
| 4 | VEC1 | "non\_existing\_file.csv" | Function raises FileNotFoundError | | Function raises FileNotFoundError | Pass | Test name: test\_EC1()  Testing a csv file that does not exist as an input |
| 5 | VEC2 | "test\_files/test\_empty.csv" | Function prints an empty string “” | | Function raises StopIteration error | Fail | Test name: test\_EC2()  Testing an empty csv file as input |
| 6 | VEC3 | “” | Function raises FileNotFoundError | | Function raises FileNotFoundError | Pass | Test name: test\_EC3()  Testing an empty string as input |
| 7 | VEC4 | "test\_files/test\_1\_column.csv" | Function prints "['Product']\n['0']\n['1']\n  ['2']\n['3']\n['4']\n" | | Function prints "['Product']\n['0']\n['1']\n  ['2']\n['3']\n['4']\n" | Pass | Test name: test\_EC4()  Testing a csv file containing only 1 column as input |
| 8 | VEC5 | "test\_files/test\_empty\_row.csv" | out[73:75] == "[]" | | out[73:75] == "[]" | Pass | Test name: test\_EC5()  Testing a csv file containing an empty row somewhere as input |
| 9 | VEC6 | "test\_files/test\_different\_types.csv" | out == "['Product']\n['Apple']\n  ['2']\n['1.0']\n['[1', '2', '3]']\n['[a', 'b', 'c]']\n" | | out == "['Product']\n['Apple']\n  ['2']\n['1.0']\n['[1', '2', '3]']\n['[a', 'b', 'c]']\n" | Pass | Test name: test\_EC6()  Testing a csv file containing different types as input |
| 10 | VEC7 | "test\_files/test\_4\_columns.csv" | out[0:39] == "['Product', 'Price', 'Units', 'Status']"  out[40:65] == "['Apple', '2', '10', '1']"  out[66:92] == "['Banana', '1', '15', '0']"  out[93:120] == "['Orange', '1.5', '8', '0']" | | out[0:39] == "['Product', 'Price', 'Units', 'Status']"  out[40:65] == "['Apple', '2', '10', '1']"  out[66:92] == "['Banana', '1', '15', '0']"  out[93:120] == "['Orange', '1.5', '8', '0']" | Pass | Test name: test\_EC7()  Testing a csv file containing 4 columns as input |
| 11 | VEC8 | "test\_files/test\_different\_column\_amounts.csv" | out[0:39] == "['Product', 'Price', 'Units', 'Status']"  out[40:49] == "['Apple']"  out[50:65] == "['Banana', '1']"  out[66:88] == "['Orange', '1.5', '8']" | | out[0:39] == "['Product', 'Price', 'Units', 'Status']"  out[40:49] == "['Apple']"  out[50:65] == "['Banana', '1']"  out[66:88] == "['Orange', '1.5', '8']" | Pass | Test name: test\_EC8()  Testing a csv file containing varying column amounts as input |
| 12 | VEC9 | "copy\_products.csv" | out[0:29]=="['Product', 'Price', 'Units']"  out[30:50] == "['Apple', '2', '10']"  out[51:72] == "['Banana', '1', '15']"  out[73:95] == "['Orange', '1.5', '8']"  out[96:116] == "['Grapes', '3', '5']" | | out[0:29]=="['Product', 'Price', 'Units']"  out[30:50] == "['Apple', '2', '10']"  out[51:72] == "['Banana', '1', '15']"  out[73:95] == "['Orange', '1.5', '8']"  out[96:116] == "['Grapes', '3', '5']" | Pass | Test name: test\_EC9()  Testing a normal flow of calling the function with a copy of products.csv |