**Test Function Document**

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Function Title: display\_filtered\_table | | | | Test Designed by: William Brunnsberg | | | |
| Test Priority (Low/Medium/High): Medium | | | | Test Designed Date: 16 November 2023 | | | |
| Description: The objective of the display\_filtered\_table function is to identify a “Product” column in the first row (header) of a csv file and then print out the rows containing a certain string provided as a parameter in the “Product”-column. | | | | 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 | VEC1 | csv\_filename: "non\_existing\_file.csv"  search: "Apple" | Function raises FileNotFoundError | | Function raises FileNotFoundError | Pass | Test name: test\_EC1()  Testing a csv file that does not exist as an csv\_filename input |
| 2 | VEC2 | csv\_filename: "test\_files/test\_empty.csv"  search: "Apple" | Function prints an empty string “” | | Function raises StopIteration error | Fail | Test name: test\_EC2()  Testing an empty csv file as csv\_filename input |
| 3 | VEC3 | csv\_filename: “”  search: "Apple" | Function raises FileNotFoundError | | Function raises FileNotFoundError | Pass | Test name: test\_EC3()  Testing an empty string as csv\_filename input |
| 4 | VEC4 | csv\_filename: "copy\_products.csv"  search: "" | out == "['Product', 'Price', 'Units']\n" | | out == "['Product', 'Price', 'Units']\n" | Pass | Test name: test\_EC4()  Testing an empty string as search input |
| 5 | VEC5 | csv\_filename: "test\_files/test\_1\_column.csv"  search: “1” | out == "['Product']\n['1']\n" | | out == "['Product']\n['1']\n" | Pass | Test name: test\_EC5()  Testing a csv file containing only 1 column as csv\_filename input |
| 6 | VEC6 | csv\_filename: "test\_files/test\_empty\_row.csv"  search: "Orange" | "['Product', 'Price', 'Units']\n['Orange', '1.5', '8']\n" | | Function raises IndexError | Fail | Test name: test\_EC6()  Testing a csv file containing an empty row somewhere as csv\_filename input |
| 7 | VEC7 | csv\_filename:  "test\_files/test\_different\_types.csv"  search: "1.0" | out == "['Product']\n['1.0']\n" | | Parsing error | Fail | Test name: test\_EC7()  Testing a csv file containing different types as csv\_filename input |
| 8 | VEC8 | csv\_filename: "test\_files/test\_4\_columns.csv"  search: "Banana" | "['Product', 'Price', 'Units', 'Status']\n['Banana', '1', '15', '0']\n" | | "['Product', 'Price', 'Units', 'Status']\n['Banana', '1', '15', '0']\n" | Pass | Test name: test\_EC8()  Testing a csv file containing 4 columns as csv\_filename input |
| 9 | VEC9 | csv\_filename: "test\_files/test\_different\_column\_amounts.csv"  search: "Banana" | out == "['Product', 'Price', 'Units', 'Status']\n['Banana', '1']\n" | | out == "['Product', 'Price', 'Units', 'Status']\n['Banana', '1']\n" | Pass | Test name: test\_EC9()  Testing a csv file containing varying column amounts as csv\_filename input |
| 10 | VEC10 | csv\_filename: "test\_files/test\_no\_product\_column.csv"  search: "Banana" | Function raises ValueError | | Function raises ValueError | Pass | Test name: test\_EC10()  Testing a csv file not containing a ‘Product’ column as csv\_filename input |
| 11 | VEC11 | csv\_filename: "test\_files/test\_product\_is\_second\_column.csv"  search: "Banana" | out == "['Price', 'Product', 'Units']\n['1', 'Banana', '15']\n" | | out == "['Price', 'Product', 'Units']\n['1', 'Banana', '15']\n" | Pass | Test name: test\_EC11()  Testing a a csv file with 'Product' as the second column as csv\_filename input |
| 12 | VEC12 | csv\_filename: "test\_files/test\_varying\_amounts\_product\_is\_second.csv"  search: "Banana" | out == "['Price', 'Product', 'Units']\n['1', 'Banana', '15']\n" | | Function raises IndexError | Fail | Test name: test\_EC12()  Testing a csv file containing varying column amounts with 'Product' as the second column as csv\_filename input |
| 13 | VEC13 | csv\_filename: "copy\_products.csv"  search: "Pancake" | out == "['Product', 'Price', 'Units']\n" | | out == "['Product', 'Price', 'Units']\n" | Pass | Test name: test\_EC13()  Testing a non-existing product as search input |
| 14 | VEC14 | csv\_filename: "copy\_products.csv"  search: "Apple" | out=="['Product', 'Price', 'Units']\n['Apple', '2', '10']\n" | | out=="['Product', 'Price', 'Units']\n['Apple', '2', '10']\n" | Pass | Test name: test\_EC14a()  Testing a normal flow with an existing product as search input and the copy\_products.csv file as csv\_filename |
| 15 | VEC14 | csv\_filename: "copy\_products.csv"  search: "Dish Soap" | out=="['Product', 'Price', 'Units']\n['Soap', '1', '12']\n['Dish Soap', '1.5', '12']\n" | | out=="['Product', 'Price', 'Units']\n['Soap', '1', '12']\n['Dish Soap', '1.5', '12']\n" | Pass | Test name: test\_EC14b()  Testing a normal flow with an existing product as search input and the copy\_products.csv file as csv\_filename |
| 16 | VEC14 | csv\_filename: "copy\_products.csv"  search: "Backpack" | out=="['Product', 'Price', 'Units']\n['Backpack', '25', '1']\n['Backpack', '15', '1']\n" | | out=="['Product', 'Price', 'Units']\n['Backpack', '25', '1']\n['Backpack', '15', '1']\n" | Pass | Test name: test\_EC14c()  Testing a normal flow with an existing product as search input and the copy\_products.csv file as csv\_filename |