Test Case Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Test Case Description | Test Data | Expected Results | Actual Results | Pass/Fail |
| SRTR-01 | Sort non-image files to the miscellaneous directory | A directory with various file types, including images and non-images | Non-image files should be moved to the specified miscellaneous directory | All non-image files were moved to the miscellaneous directory specified | PASS |
| SRTR-02 | Attempt to sort to miscellaneous when directory is empty | An empty directory | No files should be moved, and a message indicating an empty directory should be displayed | No files were moved, and an appropriate message indicating the above was printed | PASS |
| SRTR-03 | Create a copy of image files and move them to a specified directory | A directory with various image file types | Original image files left in the same directory, copied files moved to specified directory | All image files were copied and moved to the specified directory, and the original files were kept the same | PASS |
| SRTR-04 | Attempt to make a copy of image files with an empty directory | An empty directory | No files copied, and a message indicating that no files were copied | No files were copied, and an appropriate message indicating the above was printed | PASS |
| ANCH-05 | Verify that the Anchor initializes correctly with valid input values. | An input and output directory, and a sorter object | Anchor object is created successfully with the provided values. | An anchor object was created and values were correct | PASS |
| ANCH-06 | Test if the **getDirectory** method returns the correct directory path. | Anchor object with directory set to an input directory on my device | Anchor object correctly returns the input directory | Anchor object correctly returned input directory | PASS |
| ANCH-07 | Test if the **getOutputDirectory** method returns the correct output directory path. | Anchor object with directory set to an input directory on my device | Anchor object correctly returns the output directory | Anchor object correctly returned output directory | PASS |
| ANCH-08 | Test if the **getSorter** method returns the correct sorter object. | Anchor object with a sorter object passed during initialization | Sorter object reference | The Sorter object reference was printed | PASS |
| ANCH-09 | Verify that the **getID** method returns a unique identifier for each **Anchor** instance. | Two **Anchor** objects created | Different IDs for each object | Two different IDs were printed | PASS |
| SRMD-10 | Verify default constructor initializes the SortingMethod with default values. | Create SortingMethod object using the default constructor | Name: "", Tag: "", Min: 0, Max: 0, Status: DISABLED | All expected outputs were printed | PASS |
| SRMD-11 | Test parameterized constructor with valid input values. | Create SortingMethod object with name, tag, min, and max | Object is created with the provided values | All parameters were filled and SortingMethod Object had all necessary methods printed | PASS |
| SRMD-12 | Test setting the name of the SortingMethod. | SortingMethod object with an initial name, set a new name | Name is updated to the expected name | Name was updated to set name | PASS |
| SRMD-13 | Test setting the tag of the SortingMethod. | SortingMethod object with an initial tag, set a new tag | Tag is updated to the expected tag | Tag was updated to set tag | PASS |
| SRMD-14 | Test setting the minimum value of the SortingMethod. | SortingMethod object with an initial min value, set a new min value | Min value is updated to the expected min value | Min value was successfully updated | PASS |
| SRMD-15 | Test setting the maximum value of the SortingMethod. | SortingMethod object with an initial max value, set a new max value | Max value is updated to the expected max value | Max value was successfully updated | PASS |
| SRMD-16 | Test setting the status of the SortingMethod. | SortingMethod object with an initial status, set a new status | Status is updated to the expected status | Status was successfully updated | PASS |
| LCIM-17 | Verify default constructor initializes the LocalImage with default values | LocalImage object with no parameters | An empty local image object is created | An empty local image object is created | PASS |
| LCIM-18 | Test parameterized constructor with valid input values | LocalImage object with file path | A local image object is created with a file path variable | A local image object is created with a file path as a parameter | PASS |
| APPL-19 | Create an Application instance and verify the initial state. | 1. Instantiate an Application object. | The object is created with an empty list of anchors and a default timer. | An empty Application Object is created | PASS |
| APPL-20 | Add a new Anchor to the Application. | 1. Call **addAnchor** with a valid directory path, output directory path, and a Sorter instance. | A new Anchor is added to the list of anchors in the Application. | A new Anchor is added | PASS |
| APPL-21 | Remove an Anchor from the Application by index. | 1. Add an Anchor to the Application. 2. Call **removeAnchor** with the index of the added Anchor. | The specified Anchor is removed from the list of anchors. | The anchor is removed | PASS |
| APPL-22 | Remove a specific Anchor from the Application. | 1. Add an Anchor to the Application. 2. Call **removeAnchor** with the Anchor instance. | The specified Anchor is removed from the list of anchors. | The anchor is removed | PASS |
| APPL-23 | Set the timer for the Application. | 1. Call **setTimer** with a valid interval, function, and Anchor. | The timer for the Application is set with the specified interval, function, and Anchor. | The timer is set correctly | PASS |
| APPL-24 | Print the list of Anchors in the Application. | 1. Add multiple Anchors to the Application. 2. Call **printAnchors**. | The list of Anchors is printed to the console. | The list of anchors is printed | PASS |
| APPL-25 | Run the CLI application and test user input. | 1. Mock user input to simulate different CLI commands (e.g., update, start auto-sort, stop auto-sort). 2. Run the **run\_cli** method. | The application responds correctly to different user inputs. | The application works as expected, and runs as expected | PASS |
| APPL-26 | Populate the Application with a test Anchor. | 1. Call **\_debug\_populate**. | The Application is populated with a test Anchor containing sorting methods. | The Application populates with test Anchors containing sorting methods | PASS |
| APPL-27 | Set a new directory path for an existing Anchor. | 1. Add an Anchor to the Application. 2. Call **setDirectory** with a new directory path. | The directory path of the specified Anchor is updated. | The new directory path is updated on the specified anchor | PASS |
| APPL-28 | Set a new output directory path for an existing Anchor. | 1. Add an Anchor to the Application. 2. Call **setOutputDirectory** with a new output directory path. | The output directory path of the specified Anchor is updated. | The new output directory path is updated on the specified anchor | PASS |
| SHST-29 | Test the UI application using only images | A directory with only image files | The photos are sorted according to the Sorting Method | The photos were sorted to their correct output files | PASS |
| SHST-30 | Test a default case using the UI | A directory with various file types, including images and non-images | The photos are sorted according to the Sorting Method and the non-images are sorted to the miscellaneous folder | The photos were sorted to their correct output files | PASS |
| SHST-31 | Test only non-images using the UI | A directory with only non-image files | The photos are all sorted to a miscellaneous folder | The photos were sorted to the correct output file | PASS |