# **System Test Plan**

| **Instructions.** In this section, you must provide your system test plan with at least 5 test cases.  Make sure:   * You provide your sample test data * Test IDs are uniquely identified and descriptive * Test descriptions are fully specified with complete inputs, specific values, and preconditions   + Be sure to provide SPECIFIC INPUTs and VALUEs so that your test cases are repeatable * Expected results are fully specified with specific output values * All tests cover scenarios based on the problem statement * All tests cover unique scenarios for the system * All strategies for system testing are demonstrated in the tests (testing equivalence classes, testing boundary values, testing exceptions/unexpected inputs) |
| --- |

**Test Data:**

**The following text files will be used for my system test cases.**

**sample-rooms.csv**

ROOM\_ID,LENGTH,WIDTH  
Office,12,14  
Dining Room,15,15  
Living Room,25,35  
Guest Bedroom,17,16  
Guest Bathroom,10,8  
Foyer,8,8  
Kitchen,18,20

**sample-logs.csv**

TIMESTAMP,ROOM\_ID,PERCENT\_CLEANED  
06/01/2021 13:39:01,Office,78  
05/31/2021 09:27:45,Dining Room,89  
05/30/2021 10:14:41,Living Room,68  
05/28/2021 17:22:52,Living Room,70  
05/21/2021 09:16:33,Dining Room,86  
05/23/2021 18:22:11,Dining Room,89  
05/23/2021 11:51:19,Guest Bedroom,77  
05/17/2021 04:37:31,Guest Bathroom,91  
05/09/2021 18:44:23,Living Room,89  
05/12/2021 18:59:12,Living Room,94  
05/13/2021 22:20:34,Guest Bedroom,74  
05/08/2021 07:01:51,Guest Bathroom,91  
05/01/2021 10:03:11,Foyer,93  
05/03/2021 17:22:52,Living Room,92  
05/11/2021 19:00:12,Living Room,89

| **Test ID** | **Description** | **Expected Results** | **Actual Results** |
| --- | --- | --- | --- |
| **testID:**  testLoadInput  **Strategy:**  Equivalence class - loading room information and cleaning events from input files | **Preconditions:**   * CleaningManager program successfully starts, prompting the user to enter the room information input file * The files sample-rooms.csv and sample-logs.csv both exist   **Steps:**   1. User types “**sample-rooms.csv**” into the console and then presses enter 2. User types “**sample-logs.csv**” into the console and then presses enter | The sample-rooms file is successfully read, prompting the cleaning events input file to be entered.  The sample-logs text file is successfully read, and the user is then prompted to enter what type of report they would like to see.  The console displays the following (user input bolded):  Room information: **sample-rooms.csv** Cleaning events: **sample-logs.csv** Report (F,C,V,Q): | The sample-rooms and sample-logs files are both successfully read.  A prompt then prints to the console that wants input for the type of report to print.  Report (F,C,V,Q): |
| **testID:**  testFrequencyReportValid  **Strategy:**  Boundary value - enter same number of rooms as rooms are in the input file | **Preconditions:**   * CleaningManager program successfully started and has read the input files sample-rooms.csv and sample-logs.csv, prompting the user to enter the type of report they’d like to see   **Steps:**   1. User types “**F**” into the console and then presses enter 2. User types “**7**” into the console and then presses enter | The input to view a frequency report is successfully read, and the user is then prompted to enter the number of rooms to include in the report.  The input to view 7 rooms in the report is successfully read, and the frequency report is then printed into the console.  The user is then prompted again to input the type of report they would like to see.  The console displays the following, not including lines from preconditions (user input bolded):  Report (F,C,V,Q): **F** Number of rooms: **7** Frequency of Cleanings [  Living Room has been cleaned 6 times  Dining Room has been cleaned 3 times  Guest Bathroom has been cleaned 2 times  Guest Bedroom has been cleaned 2 times  Foyer has been cleaned 1 times  Office has been cleaned 1 times  Kitchen has been cleaned 0 times ] Report (F,C,V,Q): | The console asks for the number of rooms.  A frequency report is printed, listing the rooms in descending order of how many times they have been cleaned.  The console then prompts for additional input for the next report. |
| **testID:**  testFrequencyReportInvalidNegative  **Strategy:**  Exception/unexpected input - enter a negative number of rooms | **Preconditions:**   * CleaningManager program successfully started and has read the input files sample-rooms.csv and sample-logs.csv, prompting the user to enter the type of report they’d like to see   **Steps:**   1. User types “**F**” into the console and then presses enter 2. User types “**-1**” into the console and then presses enter | The input to view a frequency report is successfully read, and the user is then prompted to enter the number of rooms to include in the report.  The input to view -1 rooms in the report is successfully read and the error message is printed into the console.  The user is then prompted again to input the type of report they would like to see.  The console displays the following, not including lines from preconditions (user input bolded):  Report (F,C,V,Q): **F** Number of rooms: **-1** Number of rooms must be greater than 0. Report (F,C,V,Q): | The console asks for the number of rooms.  The console prints “Number of rooms must be greater than 0.” as an error message to the incorrect input.  The console then prompts for additional input for the next report. |
| **testID:**  testCleaningsReportInvalid  **Strategy:**  Equivalence class - selecting a cleanings report by room | **Preconditions:**   * CleaningManager program successfully started and has read the input files sample-rooms.csv and sample-logs.csv, prompting the user to enter the type of report they’d like to see   **Steps:**   1. User types “**C**” into the console and then presses enter | The input to view a cleanings report is successfully read.  The room report is then printed to the console, sorting each room in ascending alphabetical order, and within each room the cleaning dates/times sorted descending chronological order.  The user is then prompted again to input the type of report they would like to see. | A room report is printed to console, listing each room in ascending alphabetical order, and each log sorted with dates/times in descending chronological order.  The console then prompts for additional input for the next report. |
| **testID:**  testEstimatedVacuumBagLifeInvalid  **Strategy:**  Exception/unexpected input - enter an improperly formatted date | **Preconditions:**   * CleaningManager program successfully started and has read the input files sample-rooms.csv and sample-logs.csv, prompting the user to enter the type of report they’d like to see   **Steps:**   1. User types “**V**” into the console and then presses enter 2. User types “**May 5th 8PM**” and then presses enter | The input to view an estimated remaining vacuum bag life report is successfully read.  The user is then prompted to enter the date and time the vacuum was last replaced.  After inputting the incorrect format, an error message is printed to the console.  The user is then prompted again to input the type of report they would like to see.  The console displays the following, not including lines from preconditions (user input bolded):  Report (F,C,V,Q): **V** Enter the date the vacuum bag was last replaced (MM/DD/YYYY HH:MM:SS): **May 5th 8PM** Date & time must be in the format: MM/DD/YYYY HH:MM:SS Report (F,C,V,Q): | Console prompts for a date to be input.  Console prints out the error message “Date & time must be in the format: MM/DD/YYYY HH:MM:SS” as a response to the incorrect format of input.  The console then prompts for additional input for the next report. |