Name	Description	Use Case	Inputs	Expected Outputs	Fail or Success
test_file_success	Creates a mock file and tests the reading of the file in the Simulator class	Reading in a file while using the Simulator	The test file, 1000, 2000, 3000, and 4000 in the file all positive	Registers 1-4 will have values 1000-4000 respectivley	If the mentioned registers have the respective values then it is a success
test_no_file	Tests what the program will do when you input a bad file name	Bad input file name or nonexistant file	The test file	The open_file function will return False	If the mentioned function returns False then success
test_incompatible_file	Tests what the program will do when you have the wrong type of file	Wrong input file	The test file	The open_file function will return False	If the mentioned function returns False then success
test_run_success	Tests if the run function is working properly	Program needs the run function to run properly	The test file	Registers 0-4 will have values 2003, 2104, 4300, 1234, 1234 respectivley and the accumulator will have the value 1234, all postive	If the mentioned registers have the respective values then it is a success
test_run_bad_instruction_stop	Tests what the code will do the user inputs a wrong instruction	Input of bad instruction and pressing n	The test file and user inputed 'n'	Registers 0-4 will have values 2003, 8704, 4300, 1234, 0000 respectivley and the accumulator will have the value 1234, all postive	If the mentioned registers have the respective values then it is a success
test_run_bad_instruction_continue	Tests what the code will do the user inputs a wrong instruction	Input of bad instruction and pressing y	The test file and user inputed 'y'	Registers 0-5 will have values 2004, 8704, 2105, 4300, 1234, 1234 respectivley and the accumulator will have the value 1234, all postive	If the mentioned registers have the respective values then it is a success

Name	Description	Use Case	Inputs	Expected Outputs	Fail or Success
test_run_stops_at_halt	Tests what the program will do when it reaches a halt instruction	Input of halt instruction	The test file	Registers 0-4 will have values 2003, 4300, 2104, 1234, 0000 respectivley and the accumulator will have the value 1234, all postive	If the mentioned registers have the respective values then it is a success
test_report_success	Tests whether the program is creating report files at end of run time	Creation of report files after program run time	Report number	The given report number.txt files exists	If the given report file exists then it is a success
test_invalid_cancel	Tests what the function returns when it is terminated	Correct Boolean value returned when user terminates the program	User inputs 'n'	The program should return False	If the program returns False then the test is a success
test_invalid_continue	Tests if the program returns True when prompted to continue from next line	User wants to continue from the next line in the program	User inputs 'y'	The program should return True	If the program returns True then the test is a success
test_format_input_validation	Tests multlply types of bad inputs and good inputs by the User	The user inputs wrong command in the file inputted into the system	12345, hello, -123, "", +1234, +0000, -1234	The program will return -1 for bad inputs and the input for the good inputs	If all of the asserts return True then it was a success
test_format_input_formatting	Tests if the program can complete user input if the input is almost correct	The user inputs almost correct commands into the system	1234, 0000	The program will return the completed inputs +1234, +0000	If all of the asserts return True then it was a success
test_read_success	Tests if the function reads input into a desired register	The user wants to input data into a certain register	,+1000	Register 0 will have the contents of ,+1000	If the mentioned register has the respective value then it is a success
test_read_one_invalid	Tests what the program does when a good instruction and a bad instruction are given	The user inputs a bad instruction and a good instruction	hello, +1000	Register 0 will have the contents of ,+1000	If the mentioned register has the respective value then it is a success

Name	Description	Use Case	Inputs	Expected Outputs	Fail or Success
test_write_success	Tests if the program can successfully write a value to the screen	The user wants a certain value read to the screen	,+1000	The value, +1000 is read out to the console	If the value is read to the console then it was a success
test_load_success	Test if function can load a word from a register to the accumulator	The user wants to load a value into the accumulator to preform an operation	,+1000	The value , +1000 is held in the accumulator	If the value is in the accumulator then it was a success
test_store_success	Tests if the function can store words from the accumulator to a location in memory	The user wants to move a value in the accumulator into memory to use it later	,+1000	Register 0 will have the contents of ,+1000	If the mentioned register has the respective value then it is a success
test_add_success	Tests the success case for the add function	The user wants to add two numbers together	,+1234, +4321	Accumulator will hold the value , +5555	If the accumulator holds the value aforementioned then it was a success
test_add_overflow	Tests the overflow case for the add function	The user wants to add two numbers together, but the sum is too large for the amount of storage available in the system	,+5000, +5000, -5000, -5000	Add function will return False	If False is returned then it was a success
test_add_zero	Tests if the add fucntion can properly handle a zero sum	The user inputs values that add up to zero	, -1000, +1000	Accumulator will hold the value , +0000	If the accumulator holds the value aforementioned then it was a success

Name	Description	Use Case	Inputs	Expected Outputs	Fail or Success
test_add_negative	Tests the negative case for the add function	The user inputs two negative numbers to add together	, -1000, -1000	Accumulator will hold the value , -2000	If the accumulator holds the value aforementioned then it was a success
test_subtract_success	Tests the success case for the subtract function	The user wants to subtract two numbers together	,+1234, +4321	Accumulator will hold the value , +3087	If the accumulator holds the value aforementioned then it was a success
test_subtract_overflow	Tests the overflow case for the subtract function	The user wants to subtract two numbers together, but the difference is too large for the amount of storage available in the system	,-5000, +5000, +5000, -5000	Subtract function will return False	If False is returned then it was a success
test_subtract_zero	Tests if the subtraction function can handle a zero difference	The user wants to subtract two numbers together that end in a zero difference	, +1000, +1000	Accumulator will hold the value , +0000	If the accumulator holds the value aforementioned then it was a success
test_subtract_negative	Tests the negative subtract case	The user inputs two numbers that end up having a negative difference	, +1000, -1000	Accumulator will hold the value , -2000	If the accumulator holds the value aforementioned then it was a success
test_divide_success	Tests the success case for the division function	The user inputs two numbers to be divided	,+0002, +2000	Accumulator will hold the value , +1000	If the accumulator holds the value aforementioned then it was a success

Name	Description	Use Case	Inputs	Expected Outputs	Fail or Success
test_divide_zero	Tests the zero quotient case for the divide function	The user inputs two numbers where the quotient is zero	, +0001, +0000	Accumulator will hold the value , +0000	If the accumulator holds the value aforementioned then it was a success
test_divide_negative	Tests the negative quotient case for the divide function	The user inputs two numbers where the quotient is negative	, +0001, -0500	Accumulator will hold the value , -0500	If the accumulator holds the value aforementioned then it was a success
test_multiply_success	Tests the success case for the multiply function	The user inputs two numbers to be multiplied	, +0002, +2000	Accumulator will hold the value , +4000	If the accumulator holds the value aforementioned then it was a success
test_multiply overflow	Tests the overflow case for the multiply function	The user inputs two numbers where the product is too large for the memory of the system	, +0002, +5000, +0002, -5000	Multiply function will return False	If False is returned then it was a success
test_multiply_zero	Tests the zero product case for the multiply function	The user inputs two numbers where the product is zero	, +1000, +0000	Accumulator will hold the value , +0000	If the accumulator holds the value aforementioned then it was a success
test_multiply_negative	Tests the negative product case for the multiply function	The user inputs two numbers where the product is negative	, +0002, -1000	Accumulator will hold the value , -2000	If the accumulator holds the value aforementioned then it was a success

Name	Description	Use Case	Inputs	Expected Outputs	Fail or Success
test_branch_success	Tests if the system can branch to a different register address	The user wants to branch to a different place in the program	90, the desired location in memory	Current address will return 89, once the counter is incremented then it will be at 90	If the assert returns true then it was a success
test_branch_neg_move	Tests if the system will branch when the user tests for a negative number in the accumulator	The user wants to branch to a different place in the program if the number in the accumulator is negative	, -1000, 90- desired address location	Current address will return 89, once the counter is incremented then it will be at 90	If the assert returns true then it was a success
test_branch_neg_stay	Tests if the system won't branch when the user tests for a negative number in the accumulator and there isn't one	The user tests if the value in the accumulator is negative and wants to stay in the same location if it isn't	, +1000, 90- desired address location	Current address will return 0	If the assert returns true then it was a success
test_branch_zero_move	Test if the system will branch of the value in the accumulator is zero	The user wants to branch to a different place in the program if the number in the accumulator is zero	90, the desired location in memory	Current address will return 89, once the counter is incremented then it will be at 90	If the assert returns true then it was a success
test_branch_zero_stay	Tests if the system won't branch when the user tests for zero in the accumulator and there isn't one	The user tests if the value in the accumulator is zero and wants to stay in the same location if it isn't	, +1000, 90- desired address location	Current address will return 0	If the assert returns true then it was a success
test_halt_success	Tests if the function returns false to terminate the program	The user wants to halt the program	Halt instruction called	False	If the assert returns true then it was a success