Part-I:
coverage.py script is implemented to output tester cases ordered alphabetically with all of the tested methods as a list and the result was dumped in to text file, mahout-coverage.txt.

Part-II:
The total coverage strategy is implemented by selecting the maximum number of tested methods covered.
The APFD value for total coverage was calculted according the the APFD formula. The number of faults was calculated by calculating the number of faults given in the faults.txt file, the number of test cases is calculated according the number of test cases in the mahout-total-result.txt, and the Tfi sum which is the samllest number of test cases that need to be run in order to expose the fualt is calculated in the calculate_APFD() method.

The additional coverage is implemented using a type of greedy algorithm approach. This method works by using feedbacks from previous selections. It is more efficient than total coverage as it selects the maximum weight element from the space that that is not yet a part of previously selected elements. For example when applying additional greedy algorithm, if some test case A is selected first from 4 test cases namely A, B, C and D (as it covers maximum no. of methods) leaving statements methods 5 and 6 uncovered. Test case B will be skipped if it covers neither statement 6 nor statement 5. Then if Test cases C and D cover statements 5 and 6,respectively. Additional Greedy would return either A;C; D;B or A; D;C;B.
The coverage is save in mahout-additional-result.txt and the apfd value is saved in mahout-addtional-apfd.txt.
The APFD vlaue of the additional startegy was calculated the same way as total coverage but the method coverage order was different in this case.

The table below show the values of APFD for total and additional strategies. As we can see the additoinal coverage give a higher performance.

Project	**APFD/ Total Coverage**	**Additional Coverage**
mahout	0.6562816046746826	0.8102639296187684