Task 1 (a)

I considered a loop starting from the second line of the input that for each line considers whether the number is even ~~afe~~ odd. Here, each line of the input are strings which I converted to integer in order to detect whether even or odd.

Task 1 (b)

For each line, I stored their first operand, operator and second operand within variables and each time I did their mathematical calculation based on their conditioning and commanded to write it in the output file.

Task 02

In the best case scenerio the outer loop will have to traverse once only since there will not be any swapping occurances there causing to reach a time complexity of O(n). Besides, flag helps to ensure ~~weath~~ whether there is any swapping or not. This helped me to provide information for the output file.

Task 03

Selection sort minimizes the number of unnecessary swappings and I took its help in this ~~stask~~ task where I sorted (selection sort) the list of id and marks.
Then while iterating through loop, I gave a condition check within the created dictionary to make sure the marks allign to the person's id. As I found the person and ~~it's~~ his mark, I updated ~~its~~ his 'id' with '0' so that I don't end up checking it again.

Task 04

I created a list "nametime" that stored the train name, departure time and destination (to ensure cross checking). Within the loop I set conditions for lexicographical and time check that was performed through 'bubble sort' which sorted 'nametime' in order. Then with the help of looping in checked the elements within 'new' list and if it matched, I passed it towards ~~the~~ output file.