Muhammet Bozkurt
150150142

Report

For explanation code’s workflow you can top of 150150142.cpp .

Part 1:

In this part, I used **Knapsack** algorithm to optimize test suite selection problem.

Complexity of knapsack algorithm is O(n T) where “T” is total time and “n” is input size.

Does your algorithm work if the running times of the test suites are given as real numbers instead of discrete values?

Dynamic programming uses memoization which means all previous values are needed to store. By just knowing this we can deduce that using real values instead of discrete ones can consume all memory we have therefore, it does not look possibly.

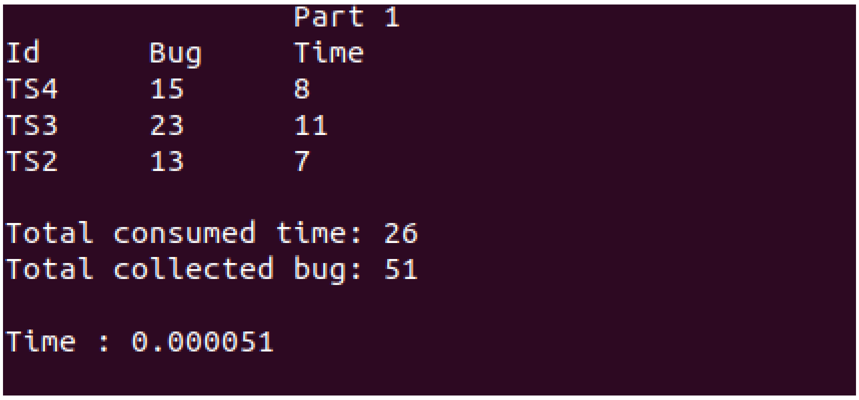
My suggestion of solving this question is approximating real values.

Mathematical Representation:

$$\text{maximize} \sum_{i=1}^{n} v_i x_i$$

$$\text{subject to} \sum_{i=1}^{n} w_i x_i \leq W \text{ and } x_i \geq 0$$

Result:



```
                    Part 1
Id        Bug       Time
TS4       15        8
TS3       23        11
TS2       13        7

Total consumed time: 26
Total collected bug: 51

Time : 0.000051
```

Part 2:

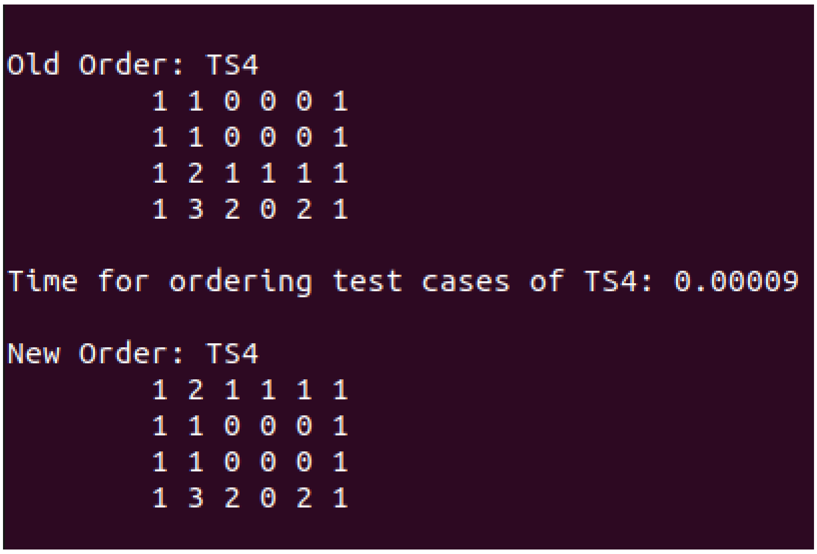
In this part, I used **Levenshtein** algorithm to measure distances between test cases.

Complexity of levenshtein algorithm is O(m x n) where “m” and “n” is input sizes but in our case test cases which we calculate distances have same size so our complexity is O(n^2).

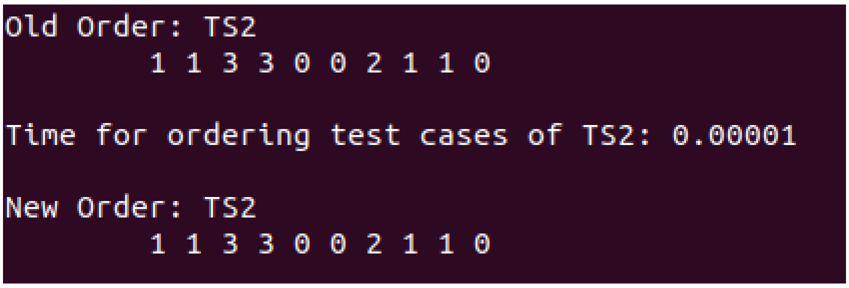
Mathematical Representation:

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j)=0, \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j)+1 \\ \operatorname{lev}_{a,b}(i,j-1)+1 \\ \operatorname{lev}_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

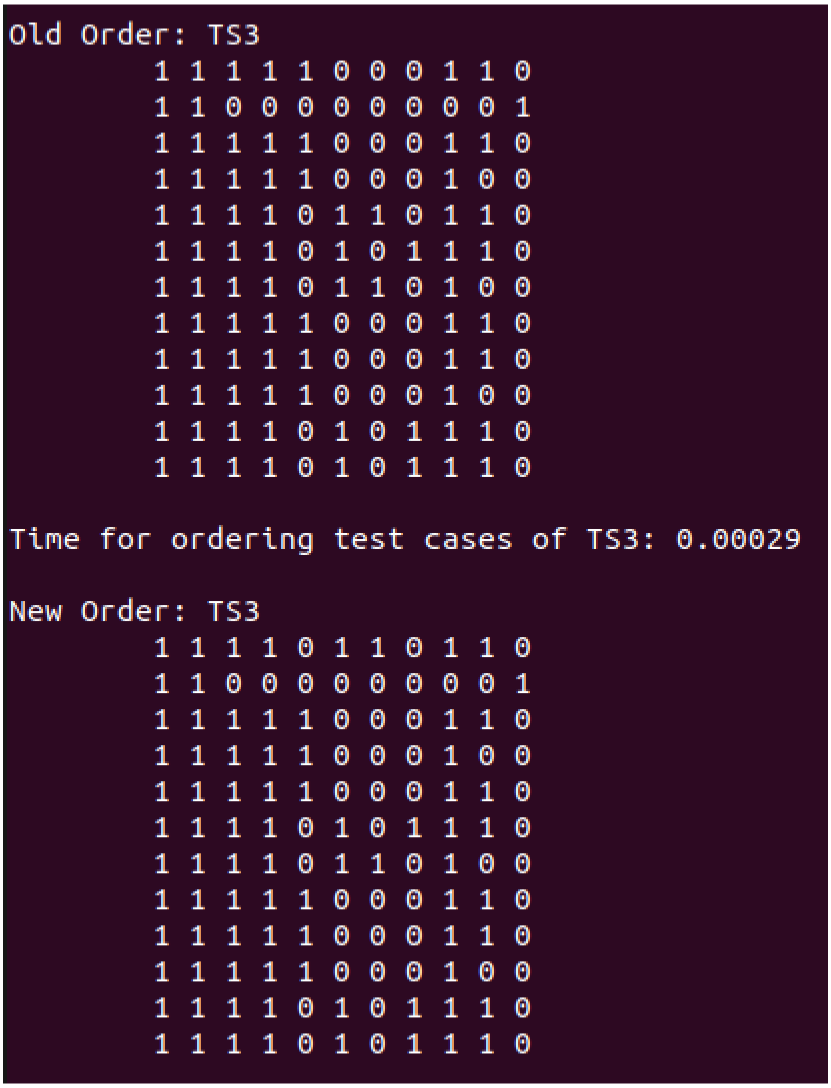
Result:



```
Old Order: TS4
        1 1 0 0 0 1
        1 1 0 0 0 1
        1 2 1 1 1 1
        1 3 2 0 2 1

Time for ordering test cases of TS4: 0.00009

New Order: TS4
        1 2 1 1 1 1
        1 1 0 0 0 1
        1 1 0 0 0 1
        1 3 2 0 2 1
```



```
Old Order: TS2
        1 1 3 3 0 0 2 1 1 0

Time for ordering test cases of TS2: 0.00001

New Order: TS2
        1 1 3 3 0 0 2 1 1 0
```



```
Old Order: TS3
        1 1 1 1 1 0 0 0 1 1 0
        1 1 0 0 0 0 0 0 0 0 1
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 1 0 0 0 1 0 0
        1 1 1 1 0 1 1 0 1 1 0
        1 1 1 1 0 1 0 1 1 1 0
        1 1 1 1 0 1 1 0 1 0 0
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 1 0 0 0 1 0 0
        1 1 1 1 0 1 0 1 1 1 0
        1 1 1 1 0 1 0 1 1 1 0

Time for ordering test cases of TS3: 0.00029

New Order: TS3
        1 1 1 1 0 1 1 0 1 1 0
        1 1 0 0 0 0 0 0 0 0 1
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 1 0 0 0 1 0 0
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 0 1 0 1 1 1 0
        1 1 1 1 0 1 1 0 1 0 0
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 1 0 0 0 1 1 0
        1 1 1 1 1 0 0 0 1 0 0
        1 1 1 1 0 1 0 1 1 1 0
        1 1 1 1 0 1 0 1 1 1 0
```