Christofides algorithm is technically a greedy approach to approximating TSP. This is because it stems from a minimum spanning tree before combining with matched pairs.

The runtime of the Christofides is O(V^3). This is because with the exception of the creation of minimum spanning tree and minimum weight perfect matching graphs, all other segments of Christofides run in either O(V) or O(E). Creating the MST takes O(E log V) when using Prim's or Kruskal's, which thus far are the two best options for this algorithm because of these low time complexities. Creating the MWPM graph is much less efficient however, running at O(V^3) time complexity using the blossom method. However, without the inclusion of this step, Christofides loses its unique characteristic where it guarantees that its solutions will be within a factor of 3/2 of the optimal solution weight.

Output of run_examples (run_test_cases was for exact solution?):

Only test1 is optimal

```
Running cs412_tsp_approx.py with input from: test_cases/test1/input.txt
23.0
a b c a
==========================================
Running cs412_tsp_approx.py with input from: test_cases/test2/input.txt
28.0
a c e d b a
==========================================
Running cs412_tsp_approx.py with input from: test_cases/test3/input.txt
33.0
a b c e j i g f h d a
==========================================
Running cs412_tsp_approx.py with input from: test_cases/test4/input.txt
36.0
a c k l d e f b j i h g a
==========================================
Running cs412_tsp_approx.py with input from: test_cases/test5/input.txt
40.0
a e f i l k m g d b c j h a
==========================================
```