Programming Assignment 5

Hill Climbing the Truck Packing Problem

Recall the truck packing problem:

You are given n integers $l_1, \ldots, l_N$, each representing the length of a package, as well as one integer C representing the capacity/length of a truck. We can pack a set of items S into one truck as long as $\sum_{i \in S} l_i \leq C$. Find the minimum number of trucks needed to fit all of the packages.

Your task is to design and implement a Hill Climbing algorithm in Python to solve the truck packing problem.

To do so, you need to:

- Decide how you want to encode states. You can see class 4 for an example for search algorithms, but recall that in the hill climbing algorithm each state should be a full solution, not a partial solution.
- Implement a method that generates a random state.
- Decide what you want the local neighborhood to be. Note that it might be easier to think about local transformations than the local neighborhood, i.e., operations to apply to a state to modify it.
- Unlike n-queens, you don't quite know when you are done. So your implementation should run for a fixed amount of time (e.g., 20 seconds) and at the end report the best solution (i.e., the one that uses the least number of trucks). The function `time.perf_counter()` from the `time` library might help you run something for a fixed number of seconds (look up what it does).

This might be the largest program you have ever written, so try to be organized. Use methods to define different components of your solution. For example, it might be helpful to have a main function that runs the main logic of hill climbing, one that generates a random state, one that calculates the score of a state, one that looks at all the states in the local neighborhood and picks the best, and perhaps a few others.

Sample 1 Input

```
l = [10, 2, 2, 3, 17, 9, 5, 4, 4, 5, 6, 8]
C = 17
```

Sample 1 Output

```
5
```

Sample 2 Input

```
l = [7, 5, 6, 4, 8, 2, 2, 3, 5, 4, 6, 3, 7, 1, 1, 2, 3, 4, 5, 6, 7, 1, 3, 4, 7, 3, 3]
C = 10
```

Sample 2 Output

```
12
```

Sample 3 Input

```
l = [1, 28, 2, 28, 29, 3, 30, 1, 27, 26, 25, 1, 2, 3, 1, 11, 18, 21, 10, 14, 13, 17, 15, 29,
29, 29]
C = 30
```

Sample 3 Output

```
15
```