Problem 1

1/1 point (ungraded)

One way of transporting cows is to always pick the **heaviest cow that will fit** onto the spaceship first. This is an example of a greedy algorithm. So if there are only 2 tons of free space on your spaceship, with one cow that's 3 tons and another that's 1 ton, the 1 ton cow will get put onto the spaceship.

Implement a greedy algorithm for transporting the cows back across space in the function `greedy_cow_transport`. The function returns a list of lists, where each inner list represents a trip and contains the names of cows taken on that trip.

Note: Make sure not to mutate the dictionary of cows that is passed in!

Assumptions:

- The order of the list of trips does not matter. That is, `[[1,2],[3,4]]` and `[[3,4],[1,2]]` are considered equivalent lists of trips.
- All the cows are between 0 and 100 tons in weight.
- All the cows have unique names.
- If multiple cows weigh the same amount, break ties arbitrarily.
- The spaceship has a cargo weight limit (in tons), which is passed into the function as a parameter.

Example:

Suppose the spaceship has a weight limit of 10 tons and the set of cows to transport is `{"Jesse": 6, "Maybel": 3, "Callie": 2, "Maggie": 5}`.

The greedy algorithm will first pick `Jesse` as the heaviest cow for the first trip. There is still space for 4 tons on the trip. Since `Maggie` will not fit on this trip, the greedy algorithm picks `Maybel`, the heaviest cow that will still fit. Now there is only 1 ton of space left, and none of the cows can fit in that space, so the first trip is `[Jesse, Maybel]`.

For the second trip, the greedy algorithm first picks `Maggie` as the heaviest remaining cow, and then picks `Callie` as the last cow. Since they will both fit, this makes the second trip `[[Maggie], [Callie]]`.

The final result then is `[["Jesse", "Maybel"], ["Maggie", "Callie"]]`.

```python
# Enter your code for the Greedy Cow Transport here
# Problem 1
def greedy_cow_transport(cows,limit=10):
    """
    Uses a greedy heuristic to determine an allocation of cows that attempts to
    minimize the number of spaceship trips needed to transport all the cows. The
    returned allocation of cows may or may not be optimal.
    The greedy heuristic should follow the following method:
    1. As long as the current trip can fit another cow, add the largest cow that will fit
        to the trip
    2. Once the trip is full, begin a new trip to transport the remaining cows
    Does not mutate the given dictionary of cows.
    Parameters:
    cows - a dictionary of name (string), weight (int) pairs
    limit - weight limit of the spaceship (an int)
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Test results

See full output

```
CORRECT
```

See full output

An "L379" Error: means that your code is taking too long to run or you have an infinite loop. Make sure you are implementing a greedy algorithm and not trying to brute force generate all possibilities (that is the next part of this problem set).

Submit

✓ Correct (1/1 point)

Problem Set 1: Problem 1

Hide Discussion

Topic: Sandbox / Problem Set 1: Problem 1

Show all posts

by recent activity

My code passes all tests successfully on my machine

HI, My code passes all tests in my local environment but the grader outputs different results. def greedy_cow_transport(cows,limit): sorted_cows = dict(sorte...

3

? not sure what's wrong with my code

Hi Team, when testing my lines of code in the terminal I retrieve the correct output, however not when adding my code into the grader. My output in the grad...

2