Assignment 4 Report

Tan Yee Jian

October 21, 2020

Contents

1 Q2 Biggest 3

- This is one of the most dubious questions of the semester, since

  1. There are many tools you can use, such as max(), min(), sorting, list mutation (`remove()`, `insert()`, `append()`)... But using any of them will defeat the purpose of the question.
  2. It's good to know that not all languages can be as expressive as python. For example, to insert an element into an array (in the heap) in C, you have to
     (a) Call `malloc()` to allocate more memory if your list runs too long.
     (b) Shift all the elements after your target one space back, using `memcpy()` or a for loop.
     (c) Finally insert your element.

     In these cases, it is more obvious that the question is testing you on implementation skills.

- The optimal method is as below:

```
def max3(seq):
 max1, max2, max3 = seq[0], seq[1], seq[2]
 # TODO: arrange such that max1 >= max2 >= max3
 for i in seq[3:]:  # definitely have to start from 3
     if i > max1:
         max1, max2, max3 = i, max1, max2
     elif i > max2:
         max1, max2, max3 = max1, i, max2
     elif i > max3:
         max1, max2, max3 = max1, max2, i
 return (max1, max2, max3)
```

- If you use other things than the first 3 values in the list as your top 3, you might fall into a trap.

    1. For example, if you use zeroes as your initial values, consider the list: `[-1, -1, -1]`, and your code will return `0,0,0`.

- If you want to assign values to multiple variables, The `a, b, c = 1, 2, 3` syntax can be useful! You can also use it to swap two values easily: `a, b = b, a`

- Bubble sort

    1. The idea of bubble sort is to swap neighbours which are not in order, ie left is greater or equal to right.
    2. You might notice that after every round of swapping shifts the **greatest** element to the rightmost end. Some people did that 3 times to get the result.
    3. This is still slower than just going through the elements **once** and collect the top 3 as per the optimal method.

2 Q4 Burger with combo

- Take note of multiple combos in an order. For example, if `burger, fries, drink = 2, 3, 2`, then there should be 2 combos and deduct **20** dollars from the bill.

3 Bonus questions

- Q5 "Irresistible Lucky Burger" was a bit tedious than Q6, I believe Q6 is more doable and everyone should try it out.
- Name and fame: one person who attempted both and did well in both; some others had the correct logic but minor mistakes here and there. Good job to everyone who tried, please keep trying!
- Idea for q5 is to first create the "middle" of the burger using `itertools.product()`, then you sandwich the buns top and bottom. This will create an exhaustive list of burgers you can make given the size of the burger. You just need to check the price and return the correct burgers after that.
- Q6 is straightforward list manipulation, but the complexity comes with the padding before and after, as well as the cases at the two ends of the lists.