# [CS 11] Prac 4a – Shaman II: Shampoo

## Problem Statement

Endugu is a shaman who resides in a dungeon hidden deep in the jungle. Unbeknownst to most, he has a day job of being a jeepney operator.

Today is a good day—there are so many passengers! They are in line waiting for a jeepney.

Endugu's task is to split the passenger queue into several groups, where each group will ride together in a jeepney.

Some jeepneys can accomodate up to $9$ passengers on both sides (that's why Endugu often shouts "shaman yan!"), so they can accommodate up to $20$ passengers overall (if we count the seats in front). However, some jeepneys may be able to accommodate less.

Given the sequence of passengers, as well as the number of passengers the jeepneys can accommodate in order, please split them up into their corresponding jeepney groups. It is okay for the last group to have less than the maximum passengers the jeepney can accommodate, if there aren't enough to reach it. However, please keep the order of the passengers the same.

It is guaranteed that there are enough seats to accommodate all passengers.

Every group should contain at least one passenger.

## Task Details

Your task is to implement a function called `jeepney_groups`. This function has two parameters:

- `passengers` — a `tuple` of `str`s denoting the sequence of passenger names in their order in the queue.
- `jeepney_counts` — a `tuple` of `int`s denoting the number of passengers each jeepney can accommodate, in order.

In particular, your function will be declared as follows:

```
def jeepney_groups(passengers, jeepney_counts):
```

Copy

The function must return a `tuple` of `tuple`s. Each `tuple` must consists of `str`s denoting a group of passengers.

## Restrictions

In this lab session, many names are banned. Here are a few of them: `zip`, `print`, `input`, `list`, `sort`, `reverse`, `type`. This is *not* an exhaustive list. (If you accidentally use a variable name that turns out to be banned, please rename it.)

However, several names have been unbanned compared to previous labs; see below.

For this problem:

- Recursion is **disallowed**. (The recursion limit has been greatly reduced.)
- Additional functions are **disallowed**.
- Comprehensions are allowed.
- `range` is allowed.
- The symbols `min`, `max`, `sum` and `sorted` are allowed.
- The source code limit is $650$.

## Example Calls

### Example 1 Function Call

```
jeepney_groups(('eiko', 'dagger', 'freya', 'amarant', 'vivi',
'steiner', 'zidane', 'quina'), (3, 4, 1))
```

Copy

### Example 1 Return Value

```
(
    ('eiko', 'dagger', 'freya'),
    ('amarant', 'vivi', 'steiner', 'zidane'),
    ('quina',),
)
```

Copy

### Example 2 Function Call

```
jeepney_groups(('eiko', 'dagger', 'freya', 'amarant', 'vivi',
'steiner', 'zidane', 'quina'), (3, 4, 20))
```

Copy

### Example 2 Return Value

```
(
    ('eiko', 'dagger', 'freya'),
    ('amarant', 'vivi', 'steiner', 'zidane'),
    ('quina',),
)
```

Copy

## Constraints

- The function `jeepney_groups` will be called at most $200$ times.
- `passengers` will have at most $50$ elements.
- Each jeepney can accommodate between $1$ and $20$ passengers.
- There are at most $20$ jeepneys.
- Each name is nonempty and consists of up to $10$ lowercase English letters.
- It is guaranteed that there are enough seats to accommodate all passengers.

## Scoring

- You get $120$ ❤️ points if you solve all test cases where:
  - all jeepneys will be fully occupied.
- You get $60$ ❤️ points if you solve all test cases.

---

## Clarifications

Report an issue

No clarifications have been made at this time.

Submit solution

**Points:** 180 (partial)
**Time limit:** 4.0s
**Memory limit:** 1G

**Author:**
kvatienza (Kevin Atienza)

**Problem type**

**Allowed languages**
~~NONE~~, py3