MORE ADTS, DICTIONARIES, AND LAMBDAS 5

COMPUTER SCIENCE 88

February 24, 2021

1 Questions

1. Write a function that takes in a sequence `s` and a function `fn` and returns a dictionary.

   The values of the dictionary are lists of elements from `s`. Each element `e` in a list should be constructed such that `fn(e)` is the same for all elements in that list. Finally, the key for each value should be `fn(e)`.

```
def group_by(s, fn):
    """
    >>> group_by([12, 23, 14, 45], lambda p: p // 10)
    {1: [12, 14], 2: [23], 4: [45]}
    >>> group_by(range(-3, 4), lambda x: x * x)
    {0: [0], 1: [-1, 1], 4: [-2, 2], 9: [-3, 3]}
    """
```

Solution:

```
    grouped = {}
    for x in s:
        key = fn(x)
        if key in grouped:
            grouped[key].append(x)
        else:
            grouped[key] = [x]
    return grouped
```

Shark Tank is a popular TV show where entrepreneurs pitch an idea to a group of investors, also known as sharks. The contestants try to convince any of the sharks to invest money in their idea.

2. Construct three data abstractions: ideas, entrepreneurs, and sharks. Ideas consist of a string name and boolean representing if the idea is good or not. Entrepreneurs consist of an idea, and an integer amount of money they are requesting. Sharks consist of a maximum amount of money they are willing to fund.

```
def make_idea(name, good):
```

Solution:
```
    return [name, good]
```

```
def get_name(idea):
```

Solution:
```
    return idea[0]
```

```
def is_good(idea):
```

Solution:
```
    return idea[1]
```

```
def make_entrepreneur(idea, money_request):
```

Solution:
```
    return [idea, money_request]
```

```
def get_idea(entrepreneur):
```

Solution:
```
    return entrepreneur[0]
```

```
def get_money_request(entrepreneur):
```

Solution:
```
    return entrepreneur[1]
```

```
def make_shark(max_funding):
```

Solution:

```
  return max_funding
```

```
def get_max_funding(shark):
```

Solution:

```
    return shark
```

3. Let's simulate a presentation by an aspiring entrepreneur. In the pitch, the entrepreneur describes an idea to a panel (a list) of sharks. If the idea is good and any of the sharks has enough money for the entrepreneur's request, the entrepreneur succeeds!

   Implement `episode`, which prints the idea of the entrepreneur and then returns `True` if any of the sharks fund it and `False` otherwise. Don't violate any data abstractions!

```
def episode(entrepreneur, shark):
    """Simulates a pitch from entrepreneur to shark
    >>> wfs = make_idea('Water-free shower', True)
    >>> nikki = make_entrepreneur(wfs, 100000)
    >>> sharks = [make_shark(100000000), make_shark(10000)]
    >>> result = episode(nikki, sharks)
    Water-free shower
    >>> result is True
    True
    """
```

Solution:

```
    idea = get_idea(entrepreneur)
    print(get_name(idea))
    for shark in sharks:
        if is_good(idea) and get_money_request(entrepreneur)
           <= get_max_funding(shark):
            return True
    return False
```

Draw the environment diagram for evaluating the following code

```
blue = 5
navy = lambda gold: blue + (gold * 2)

def f(blue):
    red = (lambda x, y: y + x)("plum", "sugar")
    return navy(3)

f(10)
```

Solution: Solution: https://tinyurl.com/1610e1x0

Draw the environment diagram for evaluating the following code

```
def anna(olaf):
    return lambda a, b: olaf or [a] * b

hans = [1]
elsa = anna(hans.append(4))
kristoff = elsa(3, 4)
```

Solution: Solution: https://tinyurl.com/14zjl83t