Faculty of Mathematical Economics

Data Structures and Algorithms

Instructor: **Nguyen Thanh Tuan**
DSEB Class of 2021 - 2024

Homework Assignment Week 10

Topic: Priority Queue and Binary Heap
Date Created: April 7, 2023

Problem 1: Contruct a Min Binary Heap

a. Implement an `Item` class and a `MinHeap` class which follows the Binary Min Heap data structure. Recommended `Item` class implementation and `MinHeap` attribute are given:

```
class Item:
    """Item in Heap"""
    def __init__(self, key, value):
        self.key = key
        self.val = value

    def __lt__(self, other):
        return self.key < other.key

    def __gt__(self, other):
        return self.key > other.key

    def __eq__(self, other):
        return self.key == other.key

class MinHeap:
    def __init__(self):
        self._data = []
```

Now your `MinHeap` class must include these following methods:

- `self.__len__` method to return the number of items in the heap.
- `self.is_empty` method to check if the heap is empty.
- `self._parent` method which takes in a positive integer k as input. k is the index of an item in `self._data`. This method returns index of item k's parent in `self._data`.
- `self._left` and `self._right` methods which take in a positive integer k as input. k is the index of an item in `self._data`. These methods return index of the left or right child of item k in the heap.

- `self._has_left` and `self._has_right` methods which take in a positive integer k as input. k is the index of an item in `self._data`. These methods check if item k has a left or right child in the heap.
- `add` method to add a new item into the heap. Note that the item must be placed at the appropriate position in the heap based on conditions of this data structure's design. If the new item's key has already existed in the heap, ask the user if they want to update the value.
- `get_min` method to return the item with minimum key in the heap.
- `remove_min` method to remove the item with minimum key in the heap.
- `max_heap_sort` method to sort the current min heap to a max heap.
- `self.__repr__` method to return a string representation of the heap.

Note: You can implement more methods to support compulsory ones.
b. Check your implementation by performing these tasks:

- Create a `MinHeap` object and add these items into the heap:

```
(2, 'E'), (7, 'A'), (5, 'S'), (1, 'S'), (0, 'D'), (8, 'U'), (9, 'B'
    ), (4, 'B'), (10, 'A')
```

- Call the method `remove_min` twice and print out the heap.
- Add the item with key 5 and value 'J' to the heap.
- Call the method `max_heap_sort` and print out the result.

Problem 2: Median of A Numeric Array

The median is the middle value in an ordered numeric 1D array. If the size of the array is even, there is no middle value and the median is the mean of two middle values.

Given a inordered array, implement a solution which utilize heap data structure to find its median. The standard time complexity for this solution is O(log N). Note that you cannot use Python built-in heap queue object, try to implement some simple binary heaps to support your solution instead.

Check your implementation with these examples:

```
array1 = [0, 2, 5, 7, 9, 4, 3]
>> expected output: 4
array1 = [0.5, -1, 3.25, -0.75, 2.5, 0]
>> expected output: 0.25
```

Optional Problem: Earning Assets

My seafood company TunaChiu which of course sells tuna is preparing for an IPO (An initial public offering (IPO) or stock launch is a public offering in which shares of a company are sold to institutional investors) at the end of next year. In order to sell a good price of shares to the investors, my company decided to work on some projects to increase capital before the IPO. As a single share-holder limited company, we have limited resources. TunaChiu intends to finish **4** distinct projects before the IPO. Your task is to help me identify the best project combo which maximizes the total capital after my company finishes all the projects.

I will give you three lists: `name`, `revenue` and `expense` where the i-th project has name `name[i]`, a net profit `revenue[i]` and an amount of capital `expense[i]` needed to start it. I cannot start a project if I do not have at least its required capital.

Initially, I have `initial_cap` amount of capital. When I start a project, my total capital is deducted an amount of the project's expense. When I finish it, I will obtain its revenue and the revenue will be added to my total capital.

Your solution must utilize heap data structure and return the name of the projects as well as the final maximized total capital. Check your implementation with this example:

```
name = ['TC1', 'TC2', 'TC3', 'TC4', 'TC5', 'TC6', 'TC7', 'TC8', 'TC9']
expense = [2, 1, 9, 5, 4, 13, 41, 39, 15]
revenue = [5, 5, 13, 10, 10, 36, 90, 79, 37]
initial_cap = 3
>> expected output: total_cap = 58
project = ['TC2', 'TC5', 'TC6', 'TC9']
```

Time complexity: O(N log N)
Space complexity: O(N)