تمرین کامپیوتری شماره ۳

دانشکده مهندسی برق و کامپیوتر

ساختمان داده - پاییز ۱۴۰۳

مدرس: **دکتر هشام فیلی و مهندس امیری**

مهلت تحویل: ۱۴۰۲/۰۳/۰۶ (۱۲ شب)

طراحان تمرین: عرفان میرشمس، سروش صحرائی، میثاق محقق

مقدمه

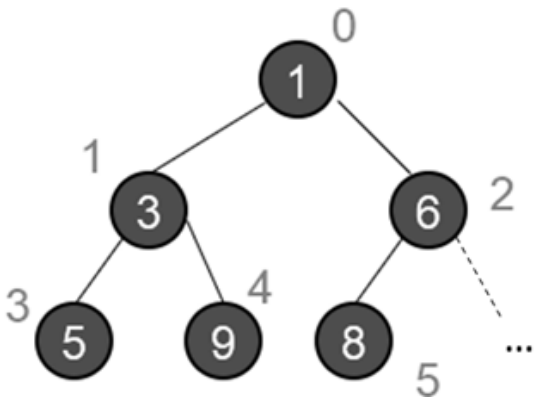
این تمرین کامپیوتری برای آشنایی با مباحث مربوط به درخت و داده ساختارهای هیپ می‌باشد. در قسمت اول به شما یک قالب از سه داده ساختار داده می‌شود و انتظار می‌رود که با توجه به مطالب گفته‌شده در رابطه با هر تابع، آنها را کامل کنید. پس از پیاده‌سازی داده ساختارهای مربوطه، در باقی سوالات به چند مسئله که با استفاده از آنها حل می‌شوند می‌پردازیم.

مسئله اول: دستگرمی (۲۵ نمره)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: میثاق محقق

توضیح داده ساختارها

min-heap: برای عناصر اضافه شده به هیپ، یک index و یک value در نظر می‌گیریم. برای درک بهتر این موضوع، در شکل زیر مقادیر داخل نودها value و مقادیر بیرون آنها index آنها هستند.



huffman-tree: دو نوع روش ورودی برای تشکیل درخت این داده ساختار داریم. نوع اول ورودی به این صورت است که لیست کاراکترها و تعداد تکرارشان داده می‌شود. نوع دوم ورودی صرفا یک متن داده می‌شود و از روی آن درخت تشکیل می‌شود.

bst: عناصر به درخت دودویی اضافه شده و سپس برخی از عملیات‌ها روی آن انجام می‌شود.

توضیح ارورها

در متدهای min-heap، حالت‌هایی وجود دارد که موجب رخ دادن ارور می‌شود (مانند pop کردن از هیپ خالی). در صورت رخ دادن حالات خاص، آنها را به صورت زیر هندل کنید:

```
raise Exception('error_text')
```

تمامی این اورها عبارتند از (بقیه اورها بررسی نمی‌شوند):

```
raise Exception('invalid index') -> ایندکس وارد شده عدد نباشد یا تایپ آن درست نباشد

raise Exception('out of range index') -> ایندکس وارد شده در محدوده سایز نباشد

raise Exception('empty') -> از هیپ خالی مقداری خارج شود
```

متن این سه ارور در ابتدای قالبِ داده شده، در متغیرهایی تعریف شده اند.

توضیح توابع

```
class MinHeap:
    class Node:
        pass

    def __init__(self): -> کانستراکتور
    def bubble_up(self, index): -> نود داده‌شده (مشخص شده با ایندکس) را تا جای ممکن به بالای هیپ می‌برد
    def bubble_down(self, index): -> نود داده‌شده (مشخص شده با ایندکس) را تا جای ممکن به پایین هیپ می‌برد
    def heap_push(self, value): -> عنصر جدید را وارد هیپ می‌کند
    def heap_pop(self): -> * روت را از هیپ خارج می‌کند و مقدارش را ریترن می‌کند
    def find_min_child(self, index): -> * ایندکس کوچکترین فرزند نود داده شده را برمی‌گرداند
    def heapify(self, *args): -> تعدادی آرگومان به عنوان ورودی دریافت کرده و آنها را وارد هیپ می‌کند


class HuffmanTree:
    class Node:
        pass

    def __init__(self): -> کانستراکتور
    def set_letters(self, *args): -> آرگومان‌های دریافتی را به عنوان حروف مورد استفاده ست می‌کند
    def set_repetitions(self, *args): -> آرگومان‌های دریافتی را به عنوان تعداد تکرار حروف ست می‌کند
    def set_text(self, text): -> حروف مورد استفاده و تعداد تکرار آنها را از روی متن ست می‌کند
    def build_tree(self): -> درخت هافمن مربوطه را می‌سازد
```

```
    def get_compressed_length(self): -> * تعداد بیت‌های متن پس از انکودینگ هافمن را برمی‌گرداند


class Bst:
    class Node:
        pass

    def __init__(self): -> کانستراکتور
    def insert(self, key): -> عنصر جدید را وارد درخت می‌کند
    def preorder(self): -> * درخت به ترتیب پیشوندی پیمایش شده و یک استرینگ از مقادیر (جدا شده با اسپیس) را برمی‌گرداند
    def inorder(self): -> * درخت به ترتیب میانوندی پیمایش شده و یک استرینگ از مقادیر (جدا شده با اسپیس) را برمی‌گرداند
    def postorder(self): -> * درخت به ترتیب پسوندی پیمایش شده و یک استرینگ از مقادیر (جدا شده با اسپیس) را برمی‌گرداند
```

نکته: توابعی که مقداری را ریترن می‌کنند با * مشخص شده اند و ممکن است الزاما نیازی به یک کلاس Node نشود.

توضیح در مورد قالب

قالب داده شده شامل چند کلاس و تابع می‌باشد و شما کافیست صرفا توابع مشخص شده در بالا را پیاده‌سازی کنید. توجه کنید که در متدهای بالا از print استفاده نکنید و مقدار مدنظر را ریترن کنید (و یا در حالت اورورها، throw کنید).

ورودی

با توجه به قالب داده شده، ابتدا یک یا چند آبجکت از نوع هیپ و درخت هافمن و درخت دودویی ایجاد می‌شود. سپس توابع مشخص شده برای هر کدام از آبجکت‌ها صدا زده می‌شوند و در صورت داشتن خروجی چاپ می‌شوند. همه توابع استفاده شده در قالب آمده است و توضیح مربوط به هر کدام در بالا آورده شده است.

توجه کنید که گرفتن و پردازش ورودی و چاپ خروجی در قالب انجام شده است و شما فقط توابع سه کلاس را کامل می‌کنید.

نمونه ورودی و خروجی ۱

```
INPUT:
make min_heap m1
call m1.heapify(10,5,30,50)
call m1.find_min_child(0)
call m1.heap_pop()
call m1.heap_pop()
call m1.heap_pop()
```

```
call m1.heap_pop()
call m1.find_min_child(-1)
call m1.find_min_child(1)
call m1.find_min_child('salap')
```

```
OUTPUT:
1
5
10
30
50
out of range index
out of range index
invalid index
```

نمونه ورودی و خروجی ۲

```
INPUT:
make bst b1
call b1.insert(50)
call b1.insert(15)
call b1.insert(20)
call b1.insert(10)
call b1.insert(40)
call b1.insert(60)
call b1.inorder()
```

```
OUTPUT:
10 15 20 40 50 60
```

نمونه ورودی و خروجی ۳

```
INPUT:
make huffman_tree h1
make huffman_tree h2
call h1.set_letters('a','b','c','d','e','f')
call h1.set_repetitions(1,3,12,13,16,1000)
call h1.build_tree()
call h1.get_compressed_length()
call h2.set_text('chaii-migholam-garm-sham-va-sard-va-tondkhoo-nabasham')
call h2.build_tree()
call h2.get_compressed_length()
```

```
OUTPUT:
1139
198
```

مسئله دوم: اسب‌های میانه (۲۵ نمره)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۶۴ مگابایت
- طراح: عرفان میرشمس

لطفعلی عاشق اسب‌ها است و می‌خواهد تعدادی اسب جدید بخرد. او همیشه اسبی با قیمت میانه را انتخاب می‌کند. به این معنی که اگر k اسب در بازار موجود باشد، لطفعلی اسبی را می‌خرد که قیمت آن در جایگاه k/2 قرار دارد. (در صورتی که تعداد اسب‌ها زوج باشد، اسب ارزان‌تر از بین دو اسب میانی را انتخاب می‌کند).

ورودی

در خط اول عدد t میاید که تعداد تست کیس های برنامه را بیان می‌کند. در خط های بعدی به اینگونه ورودی داده می‌شود: در هر خط ورودی عدد صحیح n به شما داده می‌شود اگر $n>0$ باشد این معنی را می‌دهد که اسبی با قیمت n به فروشگاه اضافه شده است. اگر n=-1 باشد این معنی را می‌دهد که لطفعلی قصد دارد از میان اسب‌های موجود خرید کند. اگر n=0 باشد این معنی را می‌دهد این تست کیس به اتمام رسیده. برنامه شما بعد از گرفتن t امین عدد صفر باید خاتمه یابد. با شروع هر تست کیس جدید هم انگار همه چیز از اول شروع شده و فروشگاه هیچ اسبی ندارد. تضمین می‌شود تعداد خط‌های ورودی از 10^5 بیشتر نمی‌شود. اعداد هم از 10^9 بزرگتر نخواهند بود.

خروجی

به ازای هر درخواست قیمت میانه (n=−1)، قیمت اسبی که لطفعلی باید بخرد را در یک خط جداگانه چاپ کنید.

نمونه ورودی و خروجی

```
INPUT:
1
9
10
2
5
1
18
```

```
-1
-1
4
3
-1
8
7
-1
0
```

```
OUTPUT:
5
9
3
7
```

مسئله سوم: نقطه روی خط (۲۵ نمره)

- محدودیت زمان: ۳ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: عرفان میرشمس

در این سوال n پاره خط روی محور مختصات به شما داده شده است. نقاط انتهایی هر پاره خط، مختصات صحیح دارند. برخی پاره‌خط‌ها ممکن است ابتدا و انتهای یکسان داشته باشند. پاره خط‌ها می‌توانند یکدیگر را قطع کنند، درون هم باشند یا حتی بر هم منطبق شوند. وظیفه شما این است: برای هر k از 1 تا n، تعداد نقاط با مختصات صحیح را محاسبه کنید که دقیقاً توسط k پاره خط پوشانده می‌شوند. نقطه x توسط پاره خط با نقاط انتهایی l_i و r_i پوشانده می‌شود اگر و تنها اگر $l_i \leq x \leq r_i$.

ورودی

خط اول ورودی شامل یک عدد صحیح $(1 \leq n \leq 2 * 10^5)$ است که تعداد پاره خط‌ها را نشان می‌دهد. n خط بعدی شامل پاره خط‌ها هستند. خط iام شامل دو عدد صحیح $(0 \leq l_i \leq r_i \leq 10^{18})$ است که نقاط انتهایی پاره خط iام را نشان می‌دهند.

خروجی

در یک خط n عدد صحیح cnt1,cnt2,⋯,cntn را با فاصله چاپ کنید که cnt_i دقیقا برابر تعداد نقاطی‌است که توسط i پاره‌خط پوشانده شده اند.

نمونه ورودی و خروجی ۱

```
INPUT:
3
```

```
0 3
1 3
3 8
```

```
OUTPUT:
6 2 1
```

نمونه ورودی و خروجی ۲

```
INPUT:
3
1 3
2 4
5 7
```

```
OUTPUT:
5 2 0
```

مسئله چهارم: اداره پست گدا (۲۵ نمره)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۱۲۸ مگابایت
- طراح: سروش صحرائی

در کشور بره، اداره پست به مشکل مالی خورده و برای فرار حل این مشکل بدترین راه حل ممکن را انتخاب کرده. اداره پست برای فرستادن هر نامه، به تعداد کاراکترهای به کار رفته از آن نامه از شهروندان پول می‌گیرد. شهروندان هم مثل همیشه با خلاقیت بسیار خود تصمیم گرفته اند جوری نامه‌هایشان را بنویسند که به اداره پست کمترین پول ممکن را بدهند. هر نامه متشکل از n کلمه $S_1, S_2, ..., S_n$ است که باید به همان ترتیب نوشته شوند.

هر کلمه از حروف کوچک و بزرگ انگلیسی و ارقام تشکیل شده. شهروندان بره برای کمتر کردن هزینه ارسال نامه، به این شکل عمل می‌کنند که ابتدا کلمه اول را در نامه می‌نویسند، سپس با شروع از کلمه دوم، به ازای هر کلمه، ابتدا بزرگترین پیشوند (prefix) یکسان آن با پسوندی (suffix) از محتوای نامه را در نظر می‌گیرند و این پیشوند را از کلمه حذف می‌کنند. سپس آن را در نامه می‌نویسند (بدون فاصله).

برای مثال یک نامه با کلمات "amir" ،"salam" به شکل "salamir" نوشته خواهد شد یا نامه‌ای با کلمات "morgh"، "riazi2" ،""ghoori" به شکل "morghooriazi2" نوشته خواهد شد.

حال شیرفرهاد که حوصله تبدیل نامه‌اش به این شکل را ندارد، از شما می‌خواهد که برای او این کار را بکنید.

ورودی

در خط اول به شما عدد n داده می‌شود که تعداد کلمه‌های نامه خواهد بود.

$$(1 \leq n \leq 10^5)$$

در خط بعدی به شما n کلمه داده می‌شود که iامین آن‌ها s_i است.

$$(\sum_{i=1}^{n} s_i \leq 10^6)$$

خروجی

نامه شیرفرهاد را به فرمت برره‌ای چاپ کنید تا اداره پست را ورشکست کنید.

نمونه ورودی و خروجی

```
INPUT:
4
Ghormeh sabzi zibaast start2
```

```
OUTPUT:
Ghormehsabzibaastart2
```

نکات تکمیلی

- هدف این تمرین یادگیری شماست. لطفاً تمرین را خودتان انجام دهید. در صورت کشف تقلب مطابق قوانین درس با آن برخورد خواهد شد.
- استفاده از کدهای آماده برای پیاده‌سازی این مباحث (جستجو شده در اینترنت و ...)، مجاز نمی‌باشد. در صورت کشف، مانند تقلب برخورد می‌شود.
- به جز سوال یک، در تمامی سوالات می‌توانید از کتابخانه‌های پیش‌فرض پایتون استفاده نمایید.
- در صورتی که تست‌های تمامی سوالات پاس شوند و نمره آنها کامل شود، ۱۰ نمره امتیازی اعمال می‌شود (نمره ۱۰۰، ۱۱۰ خواهدشد).