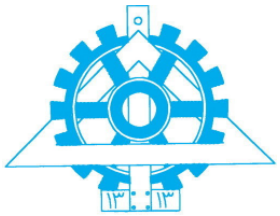


تمرین کامپیوتری شماره ۳

دانشکده مهندسی برق و کامپیوتر

ساختمان داده - پاییز ۱۴۰۲

مدرس: **دکتر هشام فیلی**

مهلت تحویل: ۱۴۰۲/۰۹/۲۸ (۱۲ شب)

طراح تمرین: ارشیا عطایی نایینی ، میثاق محقق

مقدمه

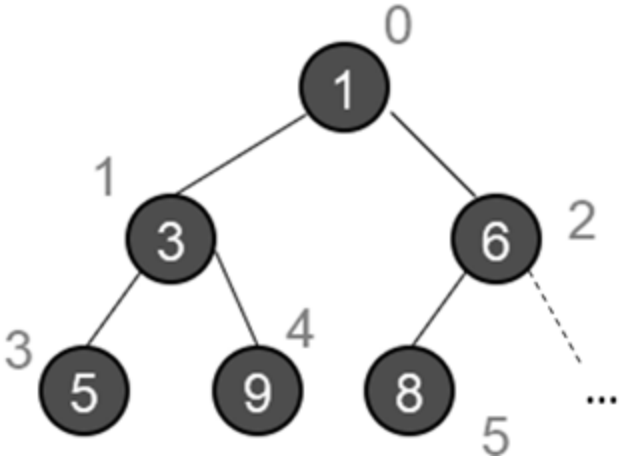
این تمرین کامپیوتری برای آشنایی با مباحث مربوط به درخت و داده ساختارهای هیپ می‌باشد. در قسمت اول به شما یک قالب از سه داده ساختار داده می‌شود و انتظار می‌رود که با توجه به مطالب گفته‌شده در رابطه با هر تابع، آنها را کامل کنید.

مسئله‌ی اول: دستگرمی (۲۵ نمره)

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

توضیح داده ساختارها:

داده ساختار min-heap : برای هر عنصری که اضافه می‌شود یک value و یک index وجود دارد که برای مثال در شکل زیر مقادیر داخل نودها، value و مقادیر بیرون آنها، index آنها هستند.



داده ساختار huffman-tree : در این قسمت به دو شکل ورودی می‌دهیم. اولی آنکه لیست کارکتر ها و تعداد تکرارشان را ست می‌کنیم. دومین روش این است که یک متن می‌دهیم. بعد از هر کدام از این دو روش، درخت مربوط به ورودی را تشکیل می‌دهیم.

داده ساختار bst : تعدادی عنصر را به آن اضافه شده و سپس تعدادی عملیات روی درخت دودویی انجام می‌شود.

توضیح اررورها :

در هر تابع، حالت هایی وجود دارد که موجب رخ دادن ارور می‌شود (مانند پاپ کردن از هیپ خالی). در صورت رخ دادن آنها، صرفا آنها را به صورت زیر هندل کنید:

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

تمام این ارور ها عبارت اند از (بقیه ارور ها 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 یا درخت خالی مقداری خارج شود
```

توضیح توابع:

```
@for_all_methods(fix_str_arg)
@for_all_methods(print_raised_exception)
class MinHeap:
    def __init__(self): -> کانستراکتور
        pass

    class Node:
        pass

    def bubble_up(self, index): -> نود داده‌شده (مشخص شده با ایندکس) را تا جای ممکن به بالای هیپ می‌برد
        pass

    def bubble_down(self, index): -> نود داده‌شده (مشخص شده با ایندکس) را تا جای ممکن به پایین هیپ می‌برد
        pass

    def heap_push(self, value): -> عنصر جدید وارد هیپ می‌شود
        pass
```

```
    def heap_pop(self): -> * روت را خارج می‌کند و مقدارش را ریترن می‌کند
        pass

    def find_min_child(self, index): -> * ایندکس کوچکترین فرزند نود داده شده را برمی‌گرداند
        pass

    def heapify(self, *args): -> تعدادی آرگومان دریافت کرده و آنها را وارد هیپ می‌کند
        pass

class HuffmanTree:
    def __init__(self): -> کانستراکتور
        pass

    @fix_str_arg
    def set_letters(self,*args): -> آرگومان های دریافتی را به‌عنوان حروف ست می‌کند
        pass

    @fix_str_arg
    def set_repetitions(self,*args): -> آرگومان های دریافتی را به‌عنوان تعداد تکرار حروف ست می‌کند
        pass

    class Node:
        pass

    def build_huffman_tree(self): -> درخت هافمن مربوطه را می‌سازد
        pass

    def get_huffman_code_cost(self): -> * هزینه انکودینگ هافمن متن داده‌شده را برمی‌گرداند
```

```
        pass

    @fix_str_arg
    def text_encoding(self, text): -> از روی متن داده شده کد هافمن را می‌سازد
        pass

@for_all_methods(fix_str_arg)
@for_all_methods(print_raised_exception)
class Bst():
    def __init__(self): -> کانستراکتور
        pass

    class Node:
        Pass

    def insert(self, key): -> عنصر جدید را وارد درخت می‌کند
        pass

    def inorder(self, key): -> درخت را به ترتیب میانوندی پیمایش و برمی‌گرداند
        pass
```

نکته : توابعی که مقداری را ریترن می‌کنند با * مشخص شده اند.

توضیح در مورد قالب

قالب شامل چند کلاس و تابع می باشد که کافی است توابع مشخص شده در بالا را کامل کنید و نیازی به یادگیری مابقی قالب نیست. در صورت صلاحدید می‌توانید متدهای دلخواه را به داده ساختارها اضافه کنید.

ورودی

با توجه به قالب داده شده ابتدا یک یا چند آبجکت از نوع پشته یا صف یا لینکد لیست ایجاد می‌شود. سپس توابع مشخص شده برای هر کدام صدا زده می‌شوند که همگی در قالب آمده است و توضیح مربوط به هرکدام در pdf تمرین آمده است.

نمونه‌ی ورودی و خروجی 1

```
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
```

نمونه‌ی ورودی و خروجی 2

```
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
```

نمونه‌ی ورودی و خروجی 3

```
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_huffman_tree()
call h1.get_huffman_code_cost()
call
h2.text_encoding('chaii-migholam-garm-sham-va-sard-va-tondkhoo-nabasham')
call h2.get_huffman_code_cost()
```

```
OUTPUT:
1139
198
```

مسئله‌ی دوم: درماندگی

- محدودیت زمان: ۱.۵ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: ارشیا عطایی نایینی

علی و پارسا دو دشمن همیشگی سعی در کلکل کردن در زمینه داده ساختار را داشتند. روزی علی برای رو کم کنی به پارسا گفت که آیا تا به‌حال اسم BST را شنیده‌ای؟ از آنجایی که پارسا طرفدار آهنگ Butter گروه BTS است، این دو را با یکدیگر اشتباه گرفت و به قاطعیت جواب مثبت به سوال علی داد. پس از آن علی گفت حالا که اینجوریه بیا این سوال را حل کن:

یک دنباله از اعداد داریم که به ترتیب در یک BST اضافه می‌شوند(اگر دو عدد برابر بودند، به بچه سمت راست اضافه می‌شود.) و در انتها تشکیل یک درخت BST می‌دهند. علی به پارسا می‌گوید به ازای هر عدد، عدد پدر آن را بگو و در انتها به ازای عدد a-ام و b-ام. اولین جد مشترک آن‌ها را بگو.(ممکن است اعداد تکراری وجود داشته باشد. شما باید به ازای هر عدد ورودی، عدد پدر آن موقع اضافه شدن را بگویید.)

از آن‌جایی که پارسا حسابی سوتی بدی داده است، از سر درماندگی به کمک شما نیاز دارد. به او کمک کنید تا جواب سوال علی را بدهد بلکه با این‌کار کمی از دشمنیشان نیز کم شود.

ورودی

در خط اول ورودی n، تعداد اعداد آمده است.$(2 \leq n \leq 1000)$
در خط بعدی n عدد آمده است که به ترتیب از چپ به راست به BST اضافه می‌شوند.

در خط بعدی دو عدد a, b آمده است که به معنای a-امین عدد ورودی و b-امین عدد ورودی است.

خروجی

در خط اول خروجی n - 1 عدد آمده است که به ترتیب شماره پدر اعداد 2 تا n ورودی هستند.

در خط بعدی شماره اولین جد مشترک اعداد a, b به ترتیب داده شدن چاپ شود.

نمونه‌ی ورودی و خروجی

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

```
OUTPUT:
3 3 5 5
1
```

مسئله‌ی سوم: کمپ (۲۵ نمره)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: ارشیا عطایی نایینی

در کمپ برنامه نویسی ایران، n مدرس وجود دارد و آموزش به مدت d روز ادامه خواهد یافت. در هر روز تنها یک ارائه داده می‌شود و مدرس i-اُم از روز A_i می‌تواند ارائه بدهد و می‌خواهد دقیقا T_i ارائه داشته باشد. یک مدرس به ازای هر تعداد کمتری که از T_i ارائه بدهد، S_i مقدار عصبانی می‌شود. حال وظیفه شما این است که جوری برنامه ارائه این d روز را مشخص کنید که کمترین میزان کلی عصبانیت را داشته باشیم.

ورودی

در خط اول ورودی d, n خواهد بود و n خط بعدی، در هر خط به ترتیب سه عدد S_i, A_i, T_i داده می‌شود که مربوط به نفر i-ام است. $(1 \leq n, d \leq 100000 \quad A, T_i \leq d \ , \ Si \leq 1e9)$

خروجی

کمترین میزان عصبانیت کلی چاپ شود.

نمونه‌ی ورودی و خروجی

```
INPUT:
2 3
1 2 300
2 2 100
```

```
OUTPUT:
100
```

مسئله‌ی چهارم: درخت بازی (۲۵ نمره)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: ارشیا عطایی نایینی

یک درخت n راسی داریم که راس 1 ریشه است و پدر راس i، P_i است. در هر راس آن، یک سکه وجود دارد که هر سکه به طور پیش‌فرض به سمت شیر خود است. در هر مرحله می‌توانیم یک راس را گرفته و سکه خود راس و همه سکه‌های داخل زیر درخت آن را پشت و رو کرد. شما باید به Q پرسمان پاسخ دهید. هر پرسمان به صورت زیر است:
مجموعه S_i به شما داده می‌شود که شامل راس‌هایی از درخت است که در ابتدا به سمت خط هستند و راس‌های خارج مجموعه به صورت شیر هستند. شما باید به ازای هر پرسمان کمترین تعداد مرحله برای اینکه در انتها همه سکه‌ها به سمت شیر خود باشند را بگویید و اگر نشدنی بود، 1- چاپ کنید.

ورودی

خط اول شامل دو عدد q, n تعداد راس‌ها و پرسمان‌ها است (اول راس‌ها بعد پرسمان). $(1 \leq n, q \leq 100000)$
خط دوم شامل n - 1 مقدار عدد می‌باشد؛ عدد i-اُم مقدار P_i است.
در Q خط بعدی، ابتدا تعداد اعضای آن مجموعه و سپس راس‌های داخل آن مجموعه می‌آیند.

خروجی

شامل Q خط خواهد بود، خط i-اُم برابر جواب این پرسمان است.

نمونه‌ی ورودی و خروجی

```
INPUT:
5 2
1 1 3 2
1 5
2 3 5
```

```
OUTPUT:
1
```

```
3
```