Data Structures and algorithms
Instructor : **Dr.Amin Eskandari**
Head-Ta's : Hamid Namjoo , Amir Hossein Hemati , Ali Mojahed
Ta's : Amir Mohammad Asadjoo , Padina razmjooi , Armin Janfaza , Alireza Heydari , Mahdi Torabi ,
Maryam Ghorbani , Fatemeh Keshavarz , MohammadReza Fasli , Abolfazl Tadrisi

Week 09 Questions

سوالات تکالیف هفته ۰۹

سوال اول (۱۰ امتیاز) : خزانه‌دار اعظم پادشاهی کسریا

در سرزمین پهناور «کسریا»، پادشاه خزانه‌ای بسیار عجیب و بی‌انتها دارد. این خزانه از nصندوقچه جادویی تشکیل شده است که در یک راهروی بسیار طولانی، پشت سر هم چیده شده‌اند و از عدد 1تا nشماره‌گذاری شده‌اند (اندیس‌های 1-based). درون هر صندوقچه تعدادی سکه طلا (یا حتی بدهی‌هایی به صورت سکه منفی!) قرار دارد.

پادشاه کسریا بسیار تنوع‌طلب و البته عجول است. او هر روز دستورات مختلفی به خزانه‌دار اعظم خود می‌دهد. گاهی اوقات دستور می‌دهد که کاروان‌های مالیاتی از راه برسند و به تمام صندوقچه‌هایی که در یک محدوده خاص قرار دارند، تعداد مشخصی سکه اضافه کنند. گاهی اوقات هم ناگهان هوس می‌کند بداند که در یک محدوده خاص از صندوقچه‌ها، مجموعاً چقدر ثروت نهفته است.

مشکل اینجاست که تعداد صندوقچه‌ها و دستورات پادشاه آن‌قدر زیاد است که خزانه‌داران قبلی به خاطر کند بودن در محاسبه، به سیاه‌چال افتاده‌اند! پادشاه برای پاسخ به هر دستور، تنها در حد یک پلک زدن ($O(\log n)$) به شما زمان می‌دهد.

شما به عنوان خزانه‌دار جدید و باهوش دربار، باید سیستم جادویی جدیدی (الگوریتم) طراحی کنید که جانتان را نجات دهد!

وظیفه شما

شما باید برنامه‌ای بنویسید که در ابتدا وضعیت اولیه nصندوقچه (یک آرایه با nعدد صحیح) را دریافت کند. سپس باید بتواند به qعملیات (درخواست‌های پادشاه) به شکل زیر پاسخ دهد:

- **نوع اول (دستور واریز):** با فرمت ADD l r v

شما باید به موجودی تمام صندوقچه‌هایی که در بازه‌ی $[l, r]$قرار دارند، دقیقاً به مقدار vسکه اضافه کنید.

- **نوع دوم (دستور گزارش‌گیری):** با فرمت SUM l r

شما باید مجموع دقیق سکه‌های موجود در بازه‌ی $[l, r]$را محاسبه کرده و به عنوان خروجی چاپ کنید.

محدودیت‌های سیستم خزانه‌داری:

- تعداد صندوقچه‌ها و تعداد درخواست‌ها: $1 \leq n, q \leq 200000$
- موجودی هر صندوقچه در هر لحظه: $|A[i]| \leq 10^9$
- شماره‌گذاری صندوقچه‌ها از یک شروع می‌شود (اندیس‌ها based-1 هستند).

شرط حیاتی پادشاه:

برای اینکه به سرنوشت خزانه‌داران قبلی دچار نشوید، طراحی شما باید به گونه‌ای باشد که **هر عملیات** (چه از نوع ADD و چه از نوع SUM) حتماً در زمان پردازش $O(\log n)$انجام شود.

نمونه ورودی و خروجی

فرمت ورودی :

ورودی شامل بخش‌های زیر است:

1. خط اول شامل دو عدد

n تعداد صندوقچه‌ها

q تعداد عملیات پادشاه

2. خط دوم شامل *n* عدد صحیح

موجودی اولیه صندوقچه‌ها (اندیس‌ها 1 هستند)

3. هر یک از خطوط بعدی، دقیقاً یکی از دو نوع دستور زیر است:

- *ADD l r v*

به همه‌ی صندوقچه‌های بازه‌ی *[l, r]* مقدار *v* اضافه می‌شود.

- *SUM l r*

مجموع موجودی صندوقچه‌های بازه‌ی [*l* , *r*] باید چاپ شود.

فرمت خروجی :

برای هر دستور از نوع *SUM*، یک عدد (مجموع محاسبه‌شده) در یک خط جداگانه چاپ می‌شود.

دستورات *ADD* خروجی ندارند.

ورودی نمونه :

```
5 4
1 2 3 4 5
SUM 1 5
ADD 2 4 10
SUM 1 5
SUM 3 3
```

خروجی نمونه :

```
15
45
13
```

سوال دوم (۳۰ امتیاز) : نقشه‌ی ستارگان و زمین‌های جادویی کسریا

پادشاه کسریا پس از اینکه با سیستم قبلی شما توانست خزانه‌ی خطی خود را مدیریت کند، به شدت تحت تاثیر هوش شما قرار گرفت! حالا او چالش بزرگ‌تری را به شما سپرده است.

پادشاه یک دشت وسیع و جادویی دارد که به صورت یک شبکه‌ی شطرنجی (گرید) دوبعدی با ابعاد $n \times m$بخش‌بندی شده است (اندیس‌ها از 1شروع می‌شوند). در ابتدا تمام این زمین‌ها بایر هستند و هیچ محصولی در آن‌ها وجود ندارد (مقدار صفر).

در طول فصل، کشاورزان در مختصات‌های خاصی از زمین بذر جادویی می‌کارند که باعث می‌شود به محصولات یک خانه مشخص افزوده شود (یا در صورت حمله آفت‌ها، از محصولات آن خانه کم شود). پادشاه که از بالکن قصرش این دشت را تماشا می‌کند، مدام دستور می‌دهد تا آمار محصولات یک محدوده‌ی مستطیل‌شکل از زمین را برایش محاسبه کنید.

از آنجا که ابعاد دشت و تعداد سوالات پادشاه بسیار زیاد است، اگر بخواهید برای هر سوال پادشاه، کل خانه‌های آن مستطیل را دانه به دانه بشمارید، روزها طول می‌کشد و پادشاه دوباره عصبانی خواهد شد! او از شما می‌خواهد که یک «نقشه جادویی» طراحی کنید که در کسری از ثانیه بتواند به سوالاتش پاسخ دهد.

وظیفه شما

شما باید یک ساختار داده پیشرفته طراحی کنید که روی این صفحه مختصات $n \times m$عمل کند و بتواند به qدستور پادشاه با فرمت‌های زیر پاسخ دهد:

- **نوع اول (دستور تغییر محصول):** با فرمت ADD x y v

شما باید به خانه‌ای که در مختصات (x, y)قرار دارد، دقیقاً به مقدار vاضافه کنید.

- **نوع دوم (دستور گزارش‌گیری مستطیلی):** با فرمت SUM x1 y1 x2 y2

شما باید مجموع تمام مقادیر (محصولات) داخل مستطیلی که گوشه‌ی بالا-چپ آن $(x1, y1)$و گوشه‌ی پایین-راست آن $(x2, y2)$است را محاسبه کرده و چاپ کنید. (دقت کنید که خود مرزهای مستطیل نیز باید در محاسبه لحاظ شوند).

محدودیت‌های دشت جادویی:

- ابعاد زمین: $1 \leq n, m \leq 1000$

- تعداد درخواست‌های پادشاه: $1 \leq q \leq 200000$
- مختصات‌ها از یک شروع می‌شوند (based-1 هستند).
- مقدار تغییرات در هر خانه: $|v| \leq 10^9$

شرط حیاتی پادشاه برای زنده ماندن شما:

برای قبولی کامل در این چالش، پیمایش کامل مستطیل با حلقه‌های تودرتو (آرایه دوبعدی ساده) به هیچ وجه قابل قبول نیست. شما باید ساختار داده‌ای طراحی کنید (که راه‌حل مورد انتظار، استفاده از **درخت فن‌ویک دوبعدی / D Fenwick Tree2** است) تا **هر عملیات** دقیقاً در زمان پردازش $O(\log n \times \log m)$انجام شود.

فرمت ورودی و خروجی :

ورودی به صورت زیر داده می‌شود:

1. خط اول سه عدد n m q

ابعاد شبکه و تعداد عملیات.

2. هر کدام از q خط بعدی یکی از این دو حالت است:
   - ADD x y v

مقدار v به خانه‌ی مختصات (x, y) اضافه می‌شود.

- SUM x1 y1 x2 y2

مجموع اعداد موجود در مستطیل با گوشه‌ی بالا-چپ (x1, y1)

و گوشه‌ی پایین-راست (x2, y2) محاسبه می‌شود.

فرمت خروجی

برای هر دستور SUM، یک عدد (پاسخ) در خطی جداگانه چاپ می‌شود.

دستورهای ADD خروجی ندارند.

ورودی نمونه :

```
3 3 6
ADD 1 1 1
ADD 1 2 2
ADD 2 1 3
SUM 1 1 2 2
SUM 2 2 3 3
SUM 1 2 1 2
```

خروجی نمونه :

```
6
0
2
```

سوال سوم (۵۰ امتیاز) :

در سرزمینی دوردست، کتابخانه‌ای باستانی و جادویی به نام «آرشیویوم» وجود داشت. در این کتابخانه، طومارهای اسرارآمیزی در یک ردیف طولانی روی قفسه‌ها چیده شده بودند. هر طومار دارای یک ارزش جادویی بود که با یک عدد صحیح نشان داده می‌شد.

محافظ این کتابخانه، جادوگری دانا به نام «آریا» بود. آریا وظیفه داشت تا نظم این طومارها را بررسی کند. گاهی اوقات، انرژی جادویی باعث می‌شد ارزش یک طومار تغییر کند. از طرفی، پادشاه سرزمین هر از گاهی پیام‌آورانی می‌فرستاد تا از آریا بپرسند در یک بخش خاص از قفسه‌ها، چقدر «بی‌نظمی» وجود

دارد. در قانون این کتابخانه، بی‌نظمی زمانی رخ می‌دهد که یک طومار با ارزش بیشتر، در سمت چپ طوماری با ارزش کمتر قرار گرفته باشد.

با گذشت زمان، تعداد طومارها و درخواست‌های پادشاه آن‌قدر زیاد شد که آریا دیگر نمی‌توانست با روش‌های معمولی به آن‌ها پاسخ دهد. او در کتاب‌های کهن خوانده بود که تنها راه مهار این حجم از جادو، ساختن یک طلسم ترکیبی و پیچیده است؛ طلسمی که از ترکیب دو ساختار قدرتمند باستانی به دست می‌آید. حالا آریا از شما، به عنوان یک برنامه‌نویس و معمار طلسم‌های منطقی، کمک خواسته است تا این سیستم را برای او پیاده‌سازی کنید.

وظیفه شما

یک آرایهٔ اولیه با nعدد صحیح داده می‌شود.

سپس باید به qعملیات پاسخ دهید.

عملیات‌ها

- UPDATE i x

مقدار عنصر در اندیس iرا برابر xقرار بده.

- QUERY l r

تعداد وارونگی‌ها (Inversions) در بازهٔ $[l, r]$را چاپ کن.

تعریف وارونگی

در بازهٔ $[l, r]$، یک جفت مرتب‌شدهٔ (i, j)وارونگی محسوب می‌شود اگر:

- $l \le i < j \le r$
- $A[i] > A[j]$

محدودیت‌ها

- $1 \leq n, q \leq 200000$
- $| A[i] | \leq 10^9$

شرط مهم طراحی

برای حل این مسئله باید به‌صورت همزمان از دو ساختار داده استفاده شود:

- یک Segment Tree روی بازه‌های آرایه
- و در هر نود آن یک Fenwick Tree (Binary Indexed Tree)

استفاده از روش‌های ساده‌تر قابل قبول نیست.

نمونه ورودی :

3 3

3 2 1

QUERY 1 3

UPDATE 1 0

QUERY 1 3

خروجی مورد انتظار :

3

1

سوال چهارم (۷۰ امتیاز) :

در اعماق کتابخانهٔ بزرگ «فرقهٔ راهبان ارغوانی»، درختی باستانی به نام «درخت دانش» وجود دارد. این درخت یک گیاه معمولی نیست؛ بلکه یک ساختار زنده برای نگهداری طلسم‌های باستانی است. راهبان برای جلوگیری از فروپاشی این درخت بر اثر سنگینی طلسم‌ها، از دو جادوی متضاد استفاده می‌کنند: جادوی **سیاه** (تثبیت‌کننده) و جادوی **قرمز** (انعطاف‌پذیر). هر بار که طلسم جدیدی به شاخه‌های درخت اضافه می‌شود، درخت با یک سری چرخش‌های پیچیده و تغییر رنگ‌ها، تعادل خود را حفظ می‌کند تا هیچ شاخه‌ای بیش از حد دراز نشود.

اخیراً، یک فرقهٔ تاریک تلاش کرده است تا با نفوذ به کتابخانه، طلسم‌های جعلی را در درخت جاسازی کند. برای مقابله با این تهدید، استادِ اعظمِ راهبان تصمیم گرفته است از «آینه‌های بی‌نقص» استفاده کند. این آینه‌ها به گونه‌ای طلسم شده‌اند که می‌توانند "جوهر" (Essence) یک شاخهٔ کامل از درخت را در قالب یک عدد یکتا خلاصه کنند.

استاد اعظم متوجه شده است که طلسم‌های جعلی همیشه به صورت شاخه‌هایی ظاهر می‌شوند که کپیِ دقیقِ یک شاخهٔ دیگر در درخت هستند. او از شما که معمار ارشد جادو هستید خواسته است تا سیستمی طراحی کنید که همزمان با رشد درخت و حفظ تعادل پیچیدهٔ آن (رنگ‌های قرمز و سیاه)، بتواند در یک چشم‌به‌هم‌زدن به او بگوید که آیا زیرشاخهٔ متصل به طلسم X، دقیقاً و موبه‌مو هم‌شکل و هم‌رنگِ زیرشاخهٔ متصل به طلسم Yهست یا خیر.

اما مراقب باشید! هر بار که طلسمی اضافه می‌شود، درخت می‌چرخد، رنگ‌ها عوض می‌شوند و جوهرِ شاخه‌ها تغییر می‌کند. شما باید آینه‌ها را طوری طراحی کنید که با هر چرخش درخت، به سرعت خود را ترمیم کنند.

وظیفه شما

شما باید یک **درخت جستجوی دودویی متعادل از نوع قرمز-سیاه (Red-Black Tree)** پیاده‌سازی کنید که در هر نود آن، علاوه بر مقدار، رنگ (قرمز یا سیاه) و اشاره‌گرها، یک **مقدار درهم‌ساز (Hash)** از کل زیردرختِ آن نود نیز ذخیره شود.

برنامه شما باید به Qکوئری پاسخ دهد. کوئری‌ها به دو دسته تقسیم می‌شوند:

1. INSERT x: مقدار صحیح xرا طبق قوانین استاندارد یک Red-Black Tree به درخت اضافه کن. پس از اضافه شدن، در صورت نیاز با چرخش‌ها (Rotations) و تغییر رنگ‌ها (Recoloring) درخت را متعادل کن. (تضمین می‌شود مقادیر ورودی تکراری نیستند).

2. CLONE_CHECK x y: دو مقدار x و y که قبلاً در درخت درج شده‌اند داده می‌شود. بگذارید U نودی باشد که مقدار x را دارد و V نودی باشد که مقدار y را دارد. شما باید بررسی کنید که آیا زیردرخت ریشه گرفته از U دقیقاً با زیردرخت ریشه گرفته از V **ایزومورف (یکسان)** است یا خیر.

شرط یکسان بودن در این مسئله: از آنجا که مقادیر نودها در درخت جستجو یکتا هستند، منظور از یکسان بودن، برابر بودن **ساختار (شکل شاخه‌ها)** و **رنگ‌بندی (قرمز/سیاه)** نودهای متناظر در دو زیردرخت است. مقادیر عددی نودها در این بررسی نادیده گرفته می‌شوند. اگر یکسان بودند عبارت YES و در غیر این صورت NO را چاپ کنید.

الزام فنی و چالش اصلی:

از آنجایی که بررسی خطی دو زیردرخت در هر کوئری منجر به خطای زمان (TLE) می‌شود، شما **باید** از تکنیک Tree Hashing (مانند Merkle Tree) استفاده کنید. هشِ یک نود باید بر اساس ترکیب ریاضیِ مقادیر زیر محاسبه شود:

- هشِ فرزند چپ
- هشِ فرزند راست
- رنگ (Color) خود نود
- تعداد کل نودهای زیردرخت (Size)

نکته بسیار مهم: در ساختار Red-Black Tree، عملیات‌های Left-Rotate و Right-Rotate ساختار پدر و فرزندی را تغییر می‌دهند. شما باید فرمول هش خود را طوری در توابع چرخش تعبیه کنید که با انجام یک چرخش در زمان $O(1)$، هش‌های نودهای درگیر نیز در زمان $O(1)$ آپدیت شوند و نیازی به پیمایش مجدد کل زیردرخت نباشد.

فرمت ورودی و خروجی

ورودی:

- در خط اول، عدد صحیح Q (تعداد کوئری‌ها) داده می‌شود ($1 \leq Q \leq 10^5$).
- در Q خط بعدی، در هر خط یکی از دو دستور INSERT x یا CLONE_CHECK x y داده می‌شود.

- محدودیت مقادیر: $1 \le x, y \le 10^9$.

خروجی:

برای هر کوئری CLONE_CHECK، با توجه به یکسان بودن کلِ زیردرخت‌ها (ساختار و رنگ‌ها)، کلمه YES یا NO را در یک خط مجزا چاپ کنید.

ورودی نمونه :

10

INSERT 10

INSERT 5

INSERT 15

INSERT 3

INSERT 7

INSERT 13

INSERT 17

CLONE_CHECK 5 15

CLONE_CHECK 3 13

CLONE_CHECK 5 10

خروجی نمونه :

YES

YES

NO

سوال پنجم (۱۰۰ امتیاز) :

شما استادکار یک کورهٔ جادویی افسانه‌ای هستید. در کارگاه شما $N = 10^5$سندان در یک ردیف قرار دارند که با شماره‌های 1تا Nمشخص شده‌اند. شما شمشیرهای قدرتمندی را می‌سازید و روی این سندان‌ها قرار می‌دهید. به دلیل محدودیت فضا، شمشیرهای جدید روی شمشیرهای قبلیِ همان سندان چیده می‌شوند (به صورت پشته).

کوره‌های جادویی بسیار داغ هستند؛ بنابراین اگر بخواهید شمشیری را ذوب کنید ، تنها می‌توانید بالاترین شمشیرِ روی یک سندان مشخص را بردارید و ذوب کنید.

همچنین پادشاه دائماً دو نوع درخواست از شما دارد: یا می‌خواهد بداند فلان شمشیر با نام خاص در حال حاضر کجاست و چقدر قدرت دارد، یا می‌خواهد بداند در بازه‌ای از سندان‌ها، قدرتمندترین شمشیری که در بالاترین نقطهٔ سندان‌ها (در دسترس‌ترین حالت) قرار دارد، چه قدرتی دارد.

جزئیات عملیات‌ها

شما باید به Qدرخواست به صورت آنلاین پاسخ دهید. درخواست‌ها شامل موارد زیر است:

1. **FORGE name power pos**: یک شمشیر جدید با نام منحصربه‌فرد name و قدرت power ساخته شده و در بالاترین نقطهٔ سندان شماره pos قرار می‌گیرد.
2. **MELT pos**: بالاترین شمشیرِ روی سندان شماره pos در کوره ذوب شده و برای همیشه از بین می‌رود (اگر سندان خالی باشد، این دستور نادیده گرفته می‌شود).
3. **INSPECT name**: پادشاه اطلاعات شمشیر name را می‌خواهد. اگر این شمشیر در حال حاضر روی هیچ‌یک از سندان‌ها نیست (یا هنوز ساخته نشده یا ذوب شده است)، عبارت NOT_FOUND را چاپ کنید. در غیر این صورت، قدرت و شمارهٔ سندان آن را چاپ کنید.
4. **MAX_TOP L R**: پادشاه می‌خواهد بداند در بین تمام سندان‌های موجود در بازهٔ $[L, R]$(شامل خود Lو R)، بیشترین قدرت متعلق به بالاترین شمشیری که روی یکی از این سندان‌ها قرار دارد، چقدر است؟ اگر تمام سندان‌های این بازه خالی باشند، عدد 0چاپ شود.

ورودی

در خط اول ورودی عدد Q(تعداد درخواست‌ها) داده می‌شود.

در Qخط بعدی، در هر خط یکی از دستورات فوق با فرمت ذکر شده داده می‌شود.

خروجی

به ازای هر دستور INSPECT و MAX_TOP، جواب مربوطه را در یک خط جداگانه چاپ کنید.

محدودیت‌ها

- $1 \leq Q \leq 10^5$
- $1 \leq pos, L, R \leq 10^5$
- $1 \leq power \leq 10^9$
- طول نام شمشیرها (رشته‌های شامل حروف کوچک انگلیسی) حداکثر 20کاراکتر است.
- تضمین می‌شود در هر لحظه، نام شمشیرهای موجود در کارگاه کاملاً منحصربه‌فرد است.

ورودی نمونه :

7

FORGE excalibur 1000 5

FORGE katana 500 5

MAX_TOP 1 10

MELT 5

MAX_TOP 1 10

INSPECT katana

INSPECT Excalibur

خروجی نمونه :

500

1000

NOT_FOUND

سوال ششم (۲۰۰ امتیاز) : ماتریس کیمیاگر و هستهٔ سرخ‌وسیاه

در آکادمی اعظم کیمیاگران، هر دانشجو برای ساخت سنگ جادو باید «ماتریس کیمیاگری» خود را مدیریت کند. این ماتریس شامل عناصر ناپایداری است که هر کدام دارای یک شناسه یکتا (ID)، قدرت تخریب ($Value$)، شماره گروه آزمایشگاهی ($Group$) و یک طلسم باستانی ($String$) هستند. به دلیل ماهیت ناپایدار این عناصر، کیمیاگر اعظم دستور داده است که سیستم ثبت عناصر باید قادر باشد در کسری از ثانیه، قدرتمندترین عنصرها را در بازه‌های مختلف پیدا کند و همچنین تغییرات لحظه‌ای گروه‌ها را زیر نظر داشته باشد.

وظیفه شما:

شما باید به عنوان مهندس ارشد آکادمی، سیستمی طراحی کنید که با دریافت Qدستور، ماتریس کیمیاگری را مدیریت کند. عملیات‌ها به شرح زیر هستند:

- ADD id val group string: یک عنصر جدید با شناسه id، ارزش val، عضویت در گروه $group$و رشته طلسم $string$به ماتریس اضافه می‌شود. این عنصر همواره در انتهای صف گروه خود قرار می‌گیرد.
- REMOVE id: عنصر با شناسه idبی‌ثبات شده و باید به طور کامل از سیستم (و از صف گروهش) حذف شود.
- MAX_VAL L R: کیمیاگر می‌خواهد بیشترین ارزش (val) در بین تمام عناصری که شناسه آن‌ها در بازه $L \le id \le R$است را بداند. (اگر هیچ عنصری در این بازه نبود، چاپ کنید NONE).
- GROUP_SUM G: محاسبه مجموع ارزش تمام عناصر فعال در گروه‌های 1تا G.
- RECENT_IN_GROUP G: یافتن ارزش (val) آخرین عنصری که به گروه Gاضافه شده و *هنوز حذف نشده است*. (اگر گروه خالی است، چاپ کنید EMPTY).

ورودی

در خط اول ورودی عدد Qمی‌آید که نشان‌دهنده تعداد عملیات‌هاست ($1 \le Q \le 10^5$).

در Qخط بعدی، در هر خط یکی از دستورات بالا با فرمت مشخص شده داده می‌شود.

- $1 \le id \le 10^9$
- $-10^9 \le val \le 10^9$
- $1 \le group \le 10^5$
- طول رشته $string$حداکثر ۱۰ کاراکتر از حروف انگلیسی است.
- تضمین می‌شود در دستور ADDشناسه‌ها (id) تکراری نیستند.

خروجی

برای هر یک از دستورات MAX_VAL، GROUP_SUM و RECENT_IN_GROUP جواب مورد نظر را در یک خط چاپ کنید.

ورودی نمونه :

6

ADD 10 500 1 alpha

ADD 20 300 2 beta

MAX_VAL 10 20

GROUP_SUM 2

RECENT_IN_GROUP 1

RECENT_IN_GROUP 3

خروجی نمونه :

500

800

500

EMPTY

سوال هفتم (۷۰ امتیاز) : بازرس ارتفاع سیاه

ما ساختار یک درخت دودویی رنگ‌آمیزی شده را به شما می‌دهیم. شما باید تعیین کنید آیا این درخت یک "درخت قرمز-سیاه" معتبر است یا خیر. یک درخت معتبر است اگر:

۱. ریشه سیاه باشد.

۲. هیچ دو گره قرمزی فرزند و والد یکدیگر نباشند.

۳. تعداد گره‌های سیاه در تمام مسیرها از هر گره به برگ‌های نیل (NULL) برابر باشد.

فرمت ورودی:

- خط اول عدد N (تعداد گره‌ها).
- در N خط بعدی، اطلاعات هر گره به صورت: id color left_child right_child می‌آید.
  - Id : شناسه گره (عدد صحیح).
  - Color : حرف Rبرای قرمز و Bبرای سیاه.
  - left_childو right_child : شناسه فرزندان (اگر 0باشد یعنی فرزندی ندارد NULL/ است).
- فرض کنید گره‌ای که id آن 1 است، همیشه ریشه است.

فرمت خروجی:

- اگر درخت معتبر است عبارت YES و در غیر این صورت NO را چاپ کنید.

ورودی نمونه:

3

1 B 2 3

2 R 0 0

3 R 0 0

خروجی نمونه:

YES

سوال هشتم (۱۰۰ امتیاز) : ساخت درخت و ریشه‌یابی

یک دنباله از اعداد به شما داده می‌شود. آن‌ها را به ترتیب در یک درخت قرمز-سیاه (که ابتدا خالی است) درج کنید. پس از اتمام تمام درج‌ها، مقدار گره ریشه و ارتفاع سیاه (Black Height)ریشه را چاپ کنید. (ارتفاع سیاه: تعداد گره‌های سیاه در مسیر از گره تا یک برگ نیل، بدون احتساب خود گره).

قوانین:

1. درج به روش استاندارد BST انجام می‌شود.
2. گره جدید همیشه قرمز است.

فرمت ورودی:

- خط اول عدد Q (تعداد اعدادی که باید درج شوند).
- خط دوم شامل Q عدد صحیح که با فاصله جدا شده‌اند.

فرمت خروجی:

- دو عدد چاپ کنید: اول مقدار گره ریشه، و دوم ارتفاع سیاه درخت.

ورودی نمونه:

3

10 20 30

خروجی نمونه:

```
20 1
```