



به نام خدا

تمرین عملی اول ساختمان داده‌ها و الگوریتم

داده ساختارهای پایه

استاد:

دکتر رستمی

استادیار:

عرشیا عموزاد

فحوهی ارسال و ارائه‌ی پاسخ تمرین

مهلت و نوع فایل‌های ارسالی:

- پاسخنامه باید شامل فایل‌های اجرایی (py) و فایل PDF شرح کدها باشد.
- مهلت ارسال: ۱۴۰۳/۰۹/۰۱ تا ساعت ۲۳:۵۹
- پاسخ‌های خود را به آیدی تلگرام Arshi82_94 بفرستید.
- روز ارائه‌ی تمرین عملی با هماهنگی در گروه تلگرامی درس اعلام خواهد شد.

ملاحظات پیاده‌سازی:

- از پیاده‌سازی آماده کتابخانه‌ها **جلوگیری** شود.
- در پیاده‌سازی های خود از داده ساختار های آموخته شده(لیست پیوندی، صف، پشته و...) استفاده کنید.
- فایل‌های template کلاس‌ها در لینک گیت هاب زیر موجود است.(به نوع خروجی‌های توابع دقت شود)
- لینک گیت‌هاب

موفق باشد

۱. مسئله ۱: تاریخچه مرورگر

شما در حال طراحی یک مرورگر با قابلیت مدیریت تاریخچه بازدیدها هستید. مرورگر در ابتدا یک صفحه اصلی (homepage) دارد و کاربر می‌تواند از آنجا به یک آدرس (url) دیگر برود، یا با استفاده از دکمه‌های Back و Forward در تاریخچه حرکت کند.

در این مسئله باید کلاس **BrowserHistory** را پیاده سازی کنید که دارای متدهای زیر می‌باشد:

۱. **BrowserHistory(homepage)**: شیء را با صفحه اصلی مرورگر مقداردهی اولیه می‌کند.

۲. **visit(url)**: کاربر از صفحه جاری به آدرس url می‌رود. تمام تاریخچه‌ی پیشرو (forward) پاک می‌شود.

۳. **back(steps)**: کاربر steps قدم به عقب بر می‌گردد. اگر فقط x قدم در تاریخچه قابل بازگشت باشد و $x > steps$ آنگاه فقط x قدم بازمی‌گردد. آدرس صفحه جاری پس از بازگشت را برگردانید.

۴. **forward(steps)**: کاربر steps قدم به جلو می‌رود. اگر فقط x قدم در تاریخچه قابل حرکت به جلو باشد و $x > steps$ آنگاه فقط x قدم به جلو می‌رود. آدرس صفحه جاری پس از حرکت به جلو را برگردانید.

* دقیق شود که مانند مرورگرهای واقعی در صورت visit جدید تاریخچه فرواردها پاک می‌شود.

ورودی:

```
["BrowserHistory", "visit", "visit", "visit", "visit", "back", "back", "forward", "visit", "forward", "back", "back"]
[["github.com"], ["google.com"], ["facebook.com"], ["youtube.com"], [1], [1], [1], ["linkedin.com"], [2], [2], [1]]
```

خروجی:

```
[null, null, null, null, "facebook.com", "google.com", "facebook.com", null, "linkedin.com", "google.com",
"github.com"]
```

۲. مسئله ۲: ساندویچ مربعی یا مستطیلی

دانشآموزان صفت کشیده‌اند. هر دانشآموز یا ساندویچ مربعی (0) را ترجیح می‌دهد یا ساندویچ مستطیلی (1). تعداد ساندویچ‌ها با تعداد دانشآموزان برابر است. ساندویچ‌ها در یک پشته (stack) قرار دارند.

فرآیند به این صورت است:

- اگر ساندویچ بالای پشته، مورد علاقه‌ی دانشآموز اول صفت باشد، آن را بر می‌دارد و صفت را ترک می‌کند.
- در غیر این صورت، دانشآموز به انتهای صفت می‌رود.

این فرآیند ادامه می‌یابد تا زمانی که هیچ دانشآموزی در صفت نماند که ساندویچ بالای پشته مورد علاقه‌اش نباشد. تعداد دانشآموزانی که غذا نخواهند خورد را برگردانید.

ورودی:

```
students = [1,1,0,0], sandwiches = [0,1,0,1]
students = [1,1,1,0,0,1], sandwiches = [1,0,0,0,1,1]
```

خروجی:

0
3

* دقیق شود در مثال‌های بالا پشته پیاده سازی نشده و در واقع ایندکس ۰ ام لیست ساندویچ‌ها بالای پشته است.

۳. مسئله ۳: دمای روزانه

یک آرایه از اعداد صحیح به شما داده می‌شود که dailyTemperatures نام دارد و نشان‌دهنده‌ی دمای روزانه است. یک آرایه برگردانید به طوری که برای هر روز در آرایه‌ی ورودی، تعداد روزهایی که باید صبر کنید تا به یک روز گرم‌تر برسید، مشخص شود. اگر چنان روزی وجود نداشت، مقدار ۰ را برای آن روز قرار دهیم.

ورودی:

temperatures = [73,74,75,71,69,72,76,73]

خروجی:

[1,1,4,2,1,1,0,0]

توضیح:

- برای روز اول (دمای ۷۳)، روز بعد گرم‌تر است (۷۴) \leftarrow ۱ روز صبر کنید.
- برای روز سوم (دمای ۷۵)، ۴ روز بعد به دمای ۷۶ می‌رسید \leftarrow ۴ روز صبر کنید.

* در پیاده‌سازی پاسخ مسئله دقت شود که حداکثر پیچیدگی زمانی $O(n)$ باشد.
راهنمایی: از یک پشته برای نگهداری روزها استفاده کنید.