



به نام خدا

# تمرین دوم ساختمان داده‌ها و الگوریتم

داده ساختارهای پایه

استاد:

دکتر روستایی

استادیار:

عرشیا عموزاد

۱. یک ماتریس اسپارس، ماتریسی است که اکثر عناصر آن صفر هستند. به عبارت دیگر، تعداد عناصر غیرصفر آن در مقایسه با کل عناصر ماتریس بسیار کم است.

ماتریس معمولی (متراکم): همه یا بیشتر خانه‌ها پر از عدد هستند.

ماتریس اسپارس (خلوت): بیشتر خانه‌ها خالی (صفر) هستند.

**تبدیل ماتریس اسپارس به متراکم**

$$\begin{bmatrix} 0 & 0 & 3 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 5 & 0 & 0 \\ 9 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 10 & 0 & -5 \end{bmatrix}$$

⇓

$$\begin{bmatrix} 5 & 6 & 7 \\ 1 & 0 & 1 \\ 1 & 5 & 8 \\ 2 & 3 & 5 \\ 3 & 0 & 9 \\ 3 & 5 & 12 \\ 4 & 3 & 10 \\ 4 & 5 & -5 \end{bmatrix}$$

سطر اول ماتریس متراکم به ترتیب تعداد سطر، تعداد ستون و تعداد درایه های غیرصفر را نشان می‌دهد.  
سطر های دیگر نیز به ترتیب شماره سطر، شماره ستون و مقدار درایه های غیر صفر ماتریس اسپارس را نشان می‌دهند.

۱. الگوریتمی برای محاسبه‌ی ترانهاده‌ی ماتریس اسپارس  $M_{p \times q}$  بنویسید.

۲. متوسط تعداد جستجو های مورد نیاز برای جستجوی دودویی موفق در ماتریس زیر چقدر است؟

$$\begin{bmatrix} 0 & 0 & 0 & 120 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 101 \\ 0 & 0 & 0 & 0 & 83 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 54 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -6 & 0 \end{bmatrix}$$

۲. برای ضرب بهینه ماتریس های زیر، آن ها را پرانتزبندی کنید.

$$A_{10 \times 2} * B_{2 \times 25} * C_{25 \times 3} * D_{3 \times 4}$$

۳. عبارت میانوندی زیر را به پسوندی تبدیل کنید.  $A * (B - D) / E - F(G + F / K)$

۴. سه پشته‌ی  $S_1, S_2, S_3$  هر یک حاوی دو عدد به شکل زیر موجود هستند.

|   |   |   |
|---|---|---|
|   |   |   |
| 1 | 3 | 5 |
| 2 | 4 | 6 |

دو عملگر  $pop(i)$  و  $poppush(i, j)$  بصورت زیر تعریف شده‌اند.  $poppush(i, j)$  یک قلم از پشته  $S_i$  حذف و به پشته  $S_j$  اضافه می‌کند.  $pop(i)$  یک قلم از پشته  $S_i$  حذف و سپس آن را چاپ می‌کند. برای چاپ اعداد ۱ تا ۶ به صورت ۱ و ۳ و ۵ و ۲ و ۴ و ۶ عملگر  $poppush$  بایستی حداقل چند بار مورد استفاده قرار گیرد؟

۵. تابع زیر چه کاری انجام می‌دهد؟ پیچیدگی زمانی آن را پیدا کنید. (ورودی تابع گره آغازین لیست پیوندی یکطرفه است)

```
1 def func(h):
2     if not h or not h.next:
3         return True
4
5     s = f = h
6     l = []
7
8     while f and f.next:
9         l.append(s.value)
10        s = s.next
11        f = f.next.next
12
13    if f:
14        s = s.next
15
16    while s:
17        if l.pop() != s.value:
18            return False
19        s = s.next
20
21    return True
```

۶. الگوریتمی ارائه دهید که بدون تغییر ساختار یک لیست پیوندی تشخیص دهد این لیست دور دارد یا خیر.

۷. عبارت  $\{x + (y - [a + b] * c - [(d + e)])\} / (j - (K - [L - n]))$  را در نظر بگیرید. می‌خواهیم با استفاده از یک پشته بررسی کنیم که آیا پرانتز، کروشه و آکولادها بدرستی تطبیق می‌شوند یا نه. پشته مورد استفاده حداقل بایستی گنجایش چند عنصر را داشته باشد؟

۴ ○

۹ ○

۱۸ ○

۲۷ ○