تمرین سری اول
ساختمان داده‌ها و الگوریتم‌ها
سررسید تحویل : ۱۳۸۵/۱۲/۲۴

---

۱. تشخیص رشته حروف:

برای هر یک از مجموعه های زیر با استفاده از تعدادی پشته (که برای هر قسمت مشخص شده است) الگوریتمی طراحی کنید که رشته ای را از ورودی دریافت کند و وجود یا عدم وجود آن رشته را در مجموعه رشته ها تعیین کند. در الگوریتم استفاده شده تنها مجاز هستید از داده های با نوع کاراکتر و بولی (boolean) استفاده کنید. (عدد بالای هر حرف نشان دهنده تعداد تکرار آن حرف است)

a) $\{a^n b^m a^m b^n \mid m, n \geq 0\}$ (1 Stack)

b) $\{a^i b^j c^k d^l \mid i+j > k+l \ \& \ i, j, k, l \geq 0\}$ (1 Stack)

c) $\{a^i b^j c^k d^l \mid i+k > j+l \ \& \ i, j, k, l \geq 0\}$ (1 Stack)

d) $\{a^{n^2} \mid n \geq 0\}$ (3 Stacks)

* در صورتی که برای بخش d بتوانید الگوریتمی با ۲ پشته ارائه دهید نمره اضافی دریافت خواهید کرد.

۲. صف ها و پشته ها:

الف) دو پشته با ظرفیت N در اختیار داریم که تنها می توانیم اعمال push و pop را روی آنها به کار بریم. الگوریتمی طراحی کنید که با استفاده از این دو پشته یک صف را شبیه سازی کند. بدین معنی که بتواند اعمال enqueue و dequeue را پشتیبانی کند. خطاهای Overflow و Underflow را نیز مد نظر داشته باشید. زمان اجرای الگوریتم را نیز تحلیل کنید.

ب) مانند بخش قبل با استفاده از دو صف یک پشته را پیاده سازی کنید. زمان اجرای الگوریتم را نیز تحلیل کنید.

۳. صف دوطرفه:

یک پشته تنها اجازه افزودن و برداشتن عناصر را از بالای خود می دهد. یک صف نیز تنها اجازه افزودن به انتها و برداشتن از ابتدا را می دهد. اما یک صف دو طرفه چنین محدودیتی ندارد و می توان عمل افزودن و برداشتن را برای ابتدا و انتهای آن به کار برد. برای چهار عمل push_front، push_back ، pop_front و pop_back در یک صف دوطرفه الگوریتم هایی طراحی کنید که در زمان O(1) اجرا شوند.

۴. یک مرتب سازی خاص:

آرایه n تایی A با اعداد متمایز ۱ تا n پر شده است. الگوریتم زیر برای مرتب کردن آرایه پیشنهاد شده است:

```
void Mysort (int []A, int n){
	int t, i;
	for (i=1; i<=n; i++)
		while (A[i] != i){
			t = A[i];
			A[i] = A[A[i]];
			A[t] = t;
		}
}
```

الف) برای اثبات درست بودن یک الگوریتم مرتب سازی باید چه گزاره هایی را اثبات کرد؟
ب) نشان دهید این الگوریتم درست کار می کند.
پ) زمان اجرای این الگوریتم را تحلیل کنید.

۵. محاسبه X^n:

الگوریتم زیر برای محاسبه X^n نوشته شده است:

```
int Pow(int x, int n) {
	if (n==0)
		return 1;
	if (n%2 == 0)
		return pow(x, n/2) * pow(x, n/2);
	else
		return pow(x, n/2) * pow(x, n/2) * x
}
```

الف) پیچیدگی زمانی الگوریتم بالا را محاسبه کنید و با نماد O نمایش دهید.
ب) با تغییر این الگوریتم زمان اجرای آن را به O(log n) تغییر دهید.
پ) الگوریتمی را که در قسمت قبل به دست آوردید به صورت غیر بازگشتی در آورید به طوری که پیچیدگی زمانی آن تغییر نکند.

۶. عنصر اکثریت:

در آرایه ای n تایی تمامی اعداد متمایز هستند به جز عدد x که $\left\lceil \frac{n}{2} \right\rceil$ بار تکرار شده است. الگوریتمی طراحی کنید که در زمان O(n) مقدار x را پیدا کند.

۷. مساله 3-3 از کتاب CLRS

توضیحات:

- پاسخ های خود را به صورت الکترونیکی و در قالب فایل PDF یا Word به آدرس ce.ds.40254@gmail.com بفرستید.
- موضوع نامه (subject) را برابر ???????HW1-8 قرار دهید به طوری که شماره دانشجویی شما به جای علامت های سوال جایگزین شده باشد. در بدنه نامه و در داخل فایل ارسالی نیز نام و نام خانوادگی خود را به همراه شماره دانشجویی ذکر کنید.
- به پاسخ های مشابه جریمه تعلق خواهد گرفت، بنابراین به شدت توصیه می‌شود که خودتان به تنهایی تمرین ها را حل کنید.
- پاسخ های شما به دقت تصحیح خواهد شد، بنابراین سعی کنید به سوالات پاسخ های محکم، مستدل و تا حد امکان کوتاه بدهید.
- مهلت ارسال پاسخ ها تا نیمه شب روز سررسید است. به پاسخ هایی که تا ۴۸ ساعت بعد از موعد فرستاده می‌شوند، به نسبت جریمه ای تعلق می‌گیرد اما تاخیر بیش از ۴۸ ساعت به هیچ وجه قابل قبول نیست. در ضمن مشکلات احتمالی سرور CE را مد نظر قرار دهید و ترجیحاً برای ارسال از سرور دیگری استفاده کنید.
- صورت تمرین ها گویا هستند، ولی در صورتی که سوالی دارید می توانید در بخش Discussion Area سایت درس سوال خود را مطرح کنید.