

Næsta stærra stak (NGE)

Bergur Snorrason

4. apríl 2022

- ▶ Látum a vera lista af n tölum.
- ▶ Okkar langar, fyrir hvert stak í listanum, að finna næst stak í listanum sem er stærra (e. *next greater element* (NGE)).
- ▶ Sem dæmi er NGE miðju stakins 4 í listanum (2, 3, 4, 8, 5) talan 8.
- ▶ Til þæginda segjum við að NGE tölunnar 8 í listanum (2, 3, 4, 8, 5) sé -1 .
- ▶ Það er auðséð að við getum reiknað NGE allra talnanna með tvöfaldri for-lykkju.

```
3 void nge(int* a, int* b, int n)
4 {
5     int i, j;
6     for (i = 0; i < n; i++)
7     {
8         for (j = 0; j < n - i; j++) if (a[i] < a[i + j]) break;
9         b[i] = (j == n - i ? -1 : i + j);
10    }
11 }
```

- ▶ Þar sem þessi lausnir er tvöföld for-lykkja, hvor af lengd n , þá er lausnin $\mathcal{O}(n^2)$.
- ▶ En þetta má bæta.

- ▶ Gefum okkur hlaða h .
- ▶ Löbbum í gegnum a í réttri röð.
- ▶ Tökum nú tölur úr hlaðan og setjum NGE þeirra talna sem a_i á meðan a_i er stærri en toppurinn á hlaðanum.
- ▶ Þegar toppurinn á hlaðanum er stærri en a_i þá látum við a_i á hlaðann og höldum svo áfram.
- ▶ Bersýnilega er hlaðinn ávallt raðaður, svo þú færð allar tölur sem eiga að hafa a_i sem NGE.
- ▶ Þegar búið er að fara í gegnum a látum við NGE þeirra staka sem eftir eru í h vera -1 .

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

|

0	1	2	3	4	5	6	7
[x	x	x	x	x	x	x	x]

h: []

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[x	x	x	x	x	x	x	x]

h: []

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[x	x	x	x	x	x	x	x]

h: [0]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[x	x	x	x	x	x	x	x]

h: [0]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
^							

0	1	2	3	4	5	6	7
[x	x	x	x	x	x	x	x]

h: [0]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
^							

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [0]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: []

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
~							

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1 2]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1 2]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
		~					

0	1	2	3	4	5	6	7
[1	x	x	x	x	x	x	x]

h: [1 2]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
		~					

0	1	2	3	4	5	6	7
[1	x	3	x	x	x	x	x]

h: [1 2]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
~							

0	1	2	3	4	5	6	7
[1	x	3	x	x	x	x	x]

h: [1]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
~							

0	1	2	3	4	5	6	7
[1	3	3	x	x	x	x	x]

h: [1]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	x	x	x	x	x]

h: []

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	x	x	x	x	x]

h: [3]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	x	x	x	x	x]

h: [3]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
			^				

0	1	2	3	4	5	6	7
[1	3	3	x	x	x	x	x]

h: [3]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
			^				

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [3]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: []

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
			~				

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
				~			

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5 6]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5 6]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
					^		

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	x	x]

h: [4 5 6]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
					^		

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	7	x]

h: [4 5 6]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
				~			

0	1	2	3	4	5	6	7
[1	3	3	4	x	x	7	x]

h: [4 5]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
				~			

0	1	2	3	4	5	6	7
[1	3	3	4	x	7	7	x]

h: [4 5]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
			~				

0	1	2	3	4	5	6	7
[1	3	3	4	x	7	7	x]

h: [4]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]
			~				

0	1	2	3	4	5	6	7
[1	3	3	4	7	7	7	x]

h: [4]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	7	7	7	x]

h: []

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	7	7	7	x]

h: [8]

0	1	2	3	4	5	6	7
[2	3	1	5	7	6	4	8]

0	1	2	3	4	5	6	7
[1	3	3	4	7	7	7	x]

h: [8]

```

3 void nge(int* a, int* b, int n)
4 {
5     int s[n], c = 0, i;
6     for (i = 0; i < n; s[c++] = i++)
7         while (c > 0 && a[s[c - 1]] < a[i]) b[s[--c]] = i;
8     while (c > 0) b[s[--c]] = -1;
9 }

```


- ▶ Við setjum hverja tölu í hlaðann að mestu einu sinni og tökum hana svo úr hlaðanum.
- ▶ Svo tímaflækjan er $\mathcal{O}(n)$.

