

Hrúgur

Bergur Snorrason

10. febrúar 2022

- ▶ Rótartvíundatré sem uppfyllir að sérhver nóða er stærri en börnin sín er sagt uppfylla *hrúguskilyrðið*.

Hrúgur

- ▶ Rótartvíundatré sem uppfyllir að sérhver nóða er stærri en börnin sín er sagt uppfylla *hrúguskilyrðið*.
- ▶ Við köllum slík tré *hrúgur* (e. *heap*).

Hrúgur

- ▶ Rótartvíundatré sem uppfyllir að sérhver nóða er stærri en börnin sín er sagt uppfylla *hrúguskilyrðið*.
- ▶ Við köllum slík tré *hrúgur* (e. *heap*).
- ▶ Hrúgur eru heppilega auðveldar í útfærslu.

Hrúgur

- ▶ Rótartvíundatré sem uppfyllir að sérhver nóða er stærri en börnin sín er sagt uppfylla *hrúguskilyrðið*.
- ▶ Við köllum slík tré *hrúgur* (e. *heap*).
- ▶ Hrúgur eru heppilega auðveldar í útfærslu.
- ▶ Við geymum tréð sem fylki og eina erfiðið er að viðhalda hrúguskilyrðinu.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Foreldri staks i er stakið $\lfloor i/2 \rfloor$.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Foreldri staks i er stakið $\lfloor i/2 \rfloor$.
- ▶ Sú seinni:

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Foreldri staks i er stakið $\lfloor i/2 \rfloor$.
- ▶ Sú seinni:
 - ▶ Rótin er í staki 0 í fylkinu.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Foreldri staks i er stakið $\lfloor i/2 \rfloor$.
- ▶ Sú seinni:
 - ▶ Rótin er í staki 0 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i + 1$.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Foreldri staks i er stakið $\lfloor i/2 \rfloor$.
- ▶ Sú seinni:
 - ▶ Rótin er í staki 0 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 2$.

- ▶ Þegar við geymum tréð sem fylki notum við eina af tveimur aðferðum.
- ▶ Sú fyrri:
 - ▶ Rótin er í staki 1 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Foreldri staks i er stakið $\lfloor i/2 \rfloor$.
- ▶ Sú seinni:
 - ▶ Rótin er í staki 0 í fylkinu.
 - ▶ Vinstra barn staks i er stak $2 \cdot i + 1$.
 - ▶ Hægra barn staks i er stak $2 \cdot i + 2$.
 - ▶ Foreldri staks i er stakið $\lfloor (i - 1)/2 \rfloor$.

- ▶ Bein afleiðing af hrúguskilyrðinu er að rótin er stærsta stakið í trénu.

- ▶ Bein afleiðing af hrúguskilyrðinu er að rótin er stærsta stakið í trénu.
- ▶ Við getum því alltaf fengið skjótan aðgang að stærsta stakinu í trénu.

- ▶ Bein afleiðing af hrúguskilyrðinu er að rótin er stærsta stakið í trénu.
- ▶ Við getum því alltaf fengið skjótan aðgang að stærsta stakinu í trénu.
- ▶ Algengt er að *forgangsbiðraðir* (e. *priority queues*) séu útfærðar með hrúgum.

```

8 #define PARENT(i) ((i - 1)/2)
9 #define LEFT(i)   ((i)*2 + 1)
10 #define RIGHT(i)  ((i)*2 + 2)
11 int h[MAXN], hn = 0;
12 void swap(int* x, int* y) { int t = *x; *x = *y; *y = t; }
13 void fix_down(int i) // Hjálparfall.
14 { // Ferðast niður tréð og lagar hrúguskilyrðið á leiðinni.
15     int mx = i;
16     if (RIGHT(i) < hn && h[mx] < h[RIGHT(i)]) mx = RIGHT(i);
17     if (LEFT(i) < hn && h[mx] < h[LEFT(i)]) mx = LEFT(i);
18     if (mx != i) swap(&h[i], &h[mx]), fix_down(mx);
19 }
20
21 void fix_up(int i) // Hjálparfall.
22 { // Ferðast upp tréð og lagar hrúguskilyrðið á leiðinni.
23     if (i == 0 || h[i] <= h[PARENT(i)]) return;
24     swap(&h[i], &h[PARENT(i)]), fix_up(PARENT(i));
25 }
26
27 void pop()
28 { // Fjarlægir stærsta stakið.
29     h[0] = h[--hn];
30     fix_down(0);
31 }
32
33 int peek() { return h[0]; } // Skilar stærsta stakinu.
34 int size() { return hn; } // Skilar stærð hrúgunar.
35 void push(int x)
36 { // Bætir x við hrúguna.
37     h[hn++] = x;
38     fix_up(hn - 1);
39 }

```

- Gerum ráð fyrir að við séum með n stök í hrúgunni.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\quad)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.
- ▶ Þar sem `pop()` þarf aðeins að ferðast einu sinni niður að laufi er tímaflækjan $\mathcal{O}(\quad)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.
- ▶ Þar sem `pop()` þarf aðeins að ferðast einu sinni niður að lafi er tímaflækjan $\mathcal{O}(\log n)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.
- ▶ Þar sem `pop()` þarf aðeins að ferðast einu sinni niður að laufi er tímaflækjan $\mathcal{O}(\log n)$.
- ▶ Þar sem `push(...)` þarf aðeins að ferðast einu sinni upp að rót er tímaflækjan $\mathcal{O}(\quad)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.
- ▶ Þar sem `pop()` þarf aðeins að ferðast einu sinni niður að laufi er tímaflækjan $\mathcal{O}(\log n)$.
- ▶ Þar sem `push(...)` þarf aðeins að ferðast einu sinni upp að rót er tímaflækjan $\mathcal{O}(\log n)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.
- ▶ Þar sem `pop()` þarf aðeins að ferðast einu sinni niður að laufi er tímaflækjan $\mathcal{O}(\log n)$.
- ▶ Þar sem `push(...)` þarf aðeins að ferðast einu sinni upp að rót er tímaflækjan $\mathcal{O}(\log n)$.
- ▶ Nú þarf `peek()` ekki að gera annað en að lesa fremsta stakið í fylki svo tímaflækjan er $\mathcal{O}(1)$.

- ▶ Gerum ráð fyrir að við séum með n stök í hrúgunni.
- ▶ Þá er hæð trésins $\mathcal{O}(\log n)$.
- ▶ Þar sem `pop()` þarf aðeins að ferðast einu sinni niður að laufi er tímaflækjan $\mathcal{O}(\log n)$.
- ▶ Þar sem `push(...)` þarf aðeins að ferðast einu sinni upp að rót er tímaflækjan $\mathcal{O}(\log n)$.
- ▶ Nú þarf `peek()` ekki að gera annað en að lesa fremsta stakið í fylki svo tímaflækjan er $\mathcal{O}(1)$.

