# Lausnir á völdum dæmum úr viku þrjú

Bergur Snorrason

2. febrúar 2022

- Ég mun leysa eftirfarandi dæmi:

- Ég mun leysa eftirfarandi dæmi:
  - *Veci*,

- Ég mun leysa eftirfarandi dæmi:
  - *Veci*,
  - *HKIO*,

- Ég mun leysa eftirfarandi dæmi:
  - *Veci*,
  - *HKIO*,
  - *Planetaris*.

# Veci

- Okkur er gefin tala $1 \leq n < 10^6$.

# Veci

- Okkur er gefin tala $1 \leq n < 10^6$.
- Hver er missta tala stærri en $n$ sem inniheldur nákvæmlega sömu tölustafi.

# Veci

- Okkur er gefin tala $1 \leq n < 10^6$.
- Hver er missta tala stærri en $n$ sem inniheldur nákvæmlega sömu tölustafi.
- Ef engin slík tala er til er svarið 0.

# Veci

- Okkur er gefin tala $1 \leq n < 10^6$.
- Hver er missta tala stærri en $n$ sem inniheldur nákvæmlega sömu tölustafi.
- Ef engin slík tala er til er svarið `0`.
- `156 -> 165`.

# Veci

- Okkur er gefin tala $1 \leq n < 10^6$.
- Hver er missta tala stærri en $n$ sem inniheldur nákvæmlega sömu tölustafi.
- Ef engin slík tala er til er svarið `0`.
- `156 -> 165`.
- `330 -> 0`.

# Veci

- Okkur er gefin tala $1 \leq n < 10^6$.
- Hver er missta tala stærri en $n$ sem inniheldur nákvæmlega sömu tölustafi.
- Ef engin slík tala er til er svarið `0`.
- `156 -> 165`.
- `330 -> 0`.
- `27711 -> 71127`.

# Veci

- Það er freystandi að reyna að útfæra gráðuga lausn.

# Veci

- Það er freystandi að reyna að útfæra gráðuga lausn.
- En þess er ekki þörf.

# Veci

- Það er freystandi að reyna að útfæra gráðuga lausn.
- En þess er ekki þörf.
- Tökum eftir að svarið hefur aldrei fleiri tölustafi en inntakið.

# Veci

- Það er freystandi að reyna að útfæra gráðuga lausn.
- En þess er ekki þörf.
- Tökum eftir að svarið hefur aldrei fleiri tölustafi en inntakið.
- Svo svarið er minna en $10^6$.

# Veci

- Það er freystandi að reyna að útfæra gráðuga lausn.
- En þess er ekki þörf.
- Tökum eftir að svarið hefur aldrei fleiri tölustafi en inntakið.
- Svo svarið er minna en $10^6$.
- Okkur nægir því að skoða allar heiltölur á bilinu $[n + 1, 10^6]$.

# Veci

- Það er freystandi að reyna að útfæra gráðuga lausn.
- En þess er ekki þörf.
- Tökum eftir að svarið hefur aldrei fleiri tölustafi en inntakið.
- Svo svarið er minna en $10^6$.
- Okkur nægir því að skoða allar heiltölur á bilinu $[n+1, 10^6]$.
- En hvernig gáum við hvort tvær tölur hafi sömu tölustafi?

# Veci

- Látum $x$ vera heiltölu.

# Veci

- Látum `x` vera heiltölu.
- Þá getum við fundið aftasta tölustafinn í `x` með `x%10`.

# Veci

- Látum `x` vera heiltölu.
- Þá getum við fundið aftasta tölustafinn í `x` með `x%10`.
- Við getum svo fjarlægt aftasta stafinn í `x` með `x/10`.

# Veci

- Látum `x` vera heiltölu.
- Þá getum við fundið aftasta tölustafinn í `x` með `x%10`.
- Við getum svo fjarlægt aftasta stafinn í `x` með `x/10`.
- Við fáum því eftirfarandi.

# Veci

```
int check(int a, int b)
{
    int c[10], i;
    for (i = 0; i < 10; i++) c[i] = 0;
    while (a > 0) c[a%10]++, a /= 10;
    while (b > 0) c[b%10]--, b /= 10;
    for (i = 0; i < 10; i++) if (c[i]) return 0;
    return 1;
}
```

# Veci

- Við þurfum nú bara að ýtra í gegnum heiltölurnar á $[n + 1, 10^6]$.

# Veci

```
int find(int n)
{
    int x = n + 1;
    while (x < 1000000 && !check(x, n)) x++;
    return x < 1000000 ? x : 0;
}
```

# Veci

- Í heildina verður þetta:

# Veci

```c
#include <stdio.h>

int check(int a, int b)
{
    int c[10], i;
    for (i = 0; i < 10; i++) c[i] = 0;
    while (a > 0) c[a%10]++, a /= 10;
    while (b > 0) c[b%10]--, b /= 10;
    for (i = 0; i < 10; i++) if (c[i]) return 0;
    return 1;
}

int find(int n)
{
    int x = n + 1;
    while (x < 1000000 && !check(x, n)) x++;
    return x < 1000000 ? x : 0;
}

int main()
{
    int n;
    scanf("%d", &n);
    printf("%d\n", find(n));
    return 0;
}
```

# Veci

- Tökum eftir að við ýtrum í gegnum $\mathcal{O}(\quad)$ tölur.

# Veci

- Tökum eftir að við ýtrum í gegnum $\mathcal{O}(n)$ tölur.

# Veci

- Tökum eftir að við ýtrum í gegnum $\mathcal{O}(\ n\ )$ tölur.
- Tímaflækjan á samanburðinum er línulegur í lengd tölustafan, það er að segja $\mathcal{O}(\qquad)$.

# Veci

- Tökum eftir að við ýtrum í gegnum $\mathcal{O}(\ n\ )$ tölur.
- Tímaflækjan á samanburðinum er línulegur í lengd tölustafan, það er að segja $\mathcal{O}(\log n)$.

# Veci

- Tökum eftir að við ýtrum í gegnum $\mathcal{O}(\ n\ )$ tölur.
- Tímaflækjan á samanburðinum er línulegur í lengd tölustafan, það er að segja $\mathcal{O}(\log n)$.
- Svo tímaflækjan í heildina er $\mathcal{O}(\qquad\qquad)$.

# Veci

- Tökum eftir að við ýtrum í gegnum $\mathcal{O}(n)$ tölur.
- Tímaflækjan á samanburðinum er línulegur í lengd tölustafan, það er að segja $\mathcal{O}(\log n)$.
- Svo tímaflækjan í heildina er $\mathcal{O}(n \log n)$.

- Gefnar eru $n$ heiltölur $a_1, \ldots, a_n$.

# HKIO

- Gefnar eru $n$ heiltölur $a_1, \dots, a_n$.
- Finnið $j \leq k$ þannig að meðaltalið $\text{avg}(a_j, a_{j+1}, \dots, a_k)$ sé hámarkað.

# HKIO

- Gefnar eru $n$ heiltölur $a_1, \ldots, a_n$.
- Finnið $j \leq k$ þannig að meðaltalið $\text{avg}(a_j, a_{j+1}, \ldots, a_k)$ sé hámarkað.
- Gefið er að $1 \leq n \leq 10^5$.

# HKIO

- Látum $m$ vera heiltölu þannig að $a_m = \max(a_1, \ldots, a_n)$.

# HKIO

- Látum $m$ vera heiltölu þannig að $a_m = \max(a_1, \ldots, a_n)$.
- Takið þá eftir að

$$\begin{aligned}
\operatorname{avg}(a_j, a_{j+1}, \ldots, a_k) &= \frac{a_j + a_{j+1} + \cdots + a_k}{j - k + 1} \\
&\leq \frac{a_m + a_m + \cdots + a_m}{j - k + 1} \\
&= \frac{(j - k + 1) \cdot a_m}{j - k + 1} \\
&= a_m.
\end{aligned}$$

# HKIO

- Látum $m$ vera heiltölu þannig að $a_m = \max(a_1, \ldots, a_n)$.
- Takið þá eftir að

$$
\begin{aligned}
\operatorname{avg}(a_j, a_{j+1}, \ldots, a_k) &= \frac{a_j + a_{j+1} + \cdots + a_k}{j - k + 1} \\
&\leq \frac{a_m + a_m + \cdots + a_m}{j - k + 1} \\
&= \frac{(j - k + 1) \cdot a_m}{j - k + 1} \\
&= a_m.
\end{aligned}
$$

- Svo meðaltalið verður aldrei stærra en $a_m$.

# HKIO

- Látum $m$ vera heiltölu þannig að $a_m = \max(a_1, \ldots, a_n)$.
- Takið þá eftir að

$$\begin{aligned} \operatorname{avg}(a_j, a_{j+1}, \ldots, a_k) &= \frac{a_j + a_{j+1} + \cdots + a_k}{j - k + 1} \\ &\leq \frac{a_m + a_m + \cdots + a_m}{j - k + 1} \\ &= \frac{(j - k + 1) \cdot a_m}{j - k + 1} \\ &= a_m. \end{aligned}$$

- Svo meðaltalið verður aldrei stærra en $a_m$.
- En einnig gildir að $\operatorname{avg}(a_m) = a_m$.

# HKIO

- Látum $m$ vera heiltölu þannig að $a_m = \max(a_1, \ldots, a_n)$.
- Takið þá eftir að

$$
\begin{aligned}
\operatorname{avg}(a_j, a_{j+1}, \ldots, a_k) &= \frac{a_j + a_{j+1} + \cdots + a_k}{j - k + 1} \\
&\leq \frac{a_m + a_m + \cdots + a_m}{j - k + 1} \\
&= \frac{(j - k + 1) \cdot a_m}{j - k + 1} \\
&= a_m.
\end{aligned}
$$

- Svo meðaltalið verður aldrei stærra en $a_m$.
- En einnig gildir að $\operatorname{avg}(a_m) = a_m$.
- Svo okkur nægir að finna stærstu töluna í listanum.

# HKIO

```c
#include <stdio.h>

int main()
{
    int i, n, r;
    scanf("%d", &n);
    int a[n];
    for (i = 0; i < n; i++) scanf("%d", &a[i]);
    for (r = i = 0; i < n; i++) if (a[i] > a[r]) r = i;
    printf("%d %d\n", r, r);
    return 0;
}
```

# Planetaris

- Atli og Finnur eru að spila tölvuleik sem snýst um að fanga sólkerfi.

# Planetaris

- Atli og Finnur eru að spila tölvuleik sem snýst um að fanga sólkerfi.
- Í leiknum eru $1 \leq n \leq 10^5$ sólkerfi.

# Planetaris

- Atli og Finnur eru að spila tölvuleik sem snýst um að fanga sólkerfi.
- Í leiknum eru $1 \leq n \leq 10^5$ sólkerfi.
- Atli og Fannar senda einhvern fjölda skipa sinna á hvert sólkerfi.

# Planetaris

- Atli og Finnur eru að spila tölvuleik sem snýst um að fanga sólkerfi.
- Í leiknum eru $1 \leq n \leq 10^5$ sólkerfi.
- Atli og Fannar senda einhvern fjölda skipa sinna á hvert sólkerfi.
- Atli fangar tiltekið sólkerfi ef hann sendir strangt fleiri skip þangað.

# Planetaris

- Atli og Finnur eru að spila tölvuleik sem snýst um að fanga sólkerfi.
- Í leiknum eru $1 \leq n \leq 10^5$ sólkerfi.
- Atli og Fannar senda einhvern fjölda skipa sinna á hvert sólkerfi.
- Atli fangar tiltekið sólkerfi ef hann sendir strangt fleiri skip þangað.
- Atli hefur $a$ skip og veit að Finnur mun senda $e_i$ skip á $i$-ta skólkerfið.

# Planetaris

- Atli og Finnur eru að spila tölvuleik sem snýst um að fanga sólkerfi.
- Í leiknum eru $1 \leq n \leq 10^5$ sólkerfi.
- Atli og Fannar senda einhvern fjölda skipa sinna á hvert sólkerfi.
- Atli fangar tiltekið sólkerfi ef hann sendir strangt fleiri skip þangað.
- Atli hefur $a$ skip og veit að Finnur mun senda $e_i$ skip á $i$-ta skólkerfið.
- Hver er mesti fjöldi sólkerfa sem Atli getur fangað?

# Planetaris

- Við græðum jafn mikið að fanga hvert sólkerfi, svo það er best að fanga þau sólkerfi sem Finnur sendir fá skip á.

# Planetaris

- Við græðum jafn mikið að fanga hvert sólkerfi, svo það er best að fanga þau sólkerfi sem Finnur sendir fá skip á.
- Við föngum því einfaldlega sólkerfin í röð, byrjum á því sem Finnur sendir fæst skip á, svo næst það sem hann sendir næst fæst skip á, og svo framvegis.

# Planetaris

- Við græðum jafn mikið að fanga hvert sólkerfi, svo það er best að fanga þau sólkerfi sem Finnur sendir fá skip á.
- Við föngum því einfaldlega sólkerfin í röð, byrjum á því sem Finnur sendir fæst skip á, svo næst það sem hann sendir næst fæst skip á, og svo framvegis.
- Þegar við föngum $i$-ta sólkerfið verðum við að passa að senda $e_i + 1$ skip, til að það verði ekki jafntefli.

# Planetaris

- Við græðum jafn mikið að fanga hvert sólkerfi, svo það er best að fanga þau sólkerfi sem Finnur sendir fá skip á.
- Við föngum því einfaldlega sólkerfin í röð, byrjum á því sem Finnur sendir fæst skip á, svo næst það sem hann sendir næst fæst skip á, og svo framvegis.
- Þegar við föngum $i$-ta sólkerfið verðum við að passa að senda $e_i + 1$ skip, til að það verði ekki jafntefli.
- Við verðum líka að passa að hætta að fanga sólkerfi þegar við höfum ekki nóg af skipum.

# Planetaris

```
int main()
{
    int i, n, a, e[MAXN];
    scanf("%d%d", &n, &a);
    for (i = 0; i < n; i++) scanf("%d", &e[i]);
    qsort(e, n, sizeof *e, cmp);
    for (i = 0; i < n; a -= e[i++] + 1) if (a < e[i] + 1) break;
    printf("%d\n", i);
    return 0;
}
```

# Planetaris

- Tímaflækjan á þessari lausn er $\mathcal{O}($ $)$ sökum .

# Planetaris

- Tímaflækjan á þessari lausn er $\mathcal{O}(n \log n)$ sökum röðunar.

# Planetaris

- Tímaflækjan á þessari lausn er $\mathcal{O}(n \log n)$ sökum röðunar.
- Takið eftir að það er mjög auðvelt að gera litlar villur sem gera lausnin ranga.

# Planetaris

- Tímaflækjan á þessari lausn er $\mathcal{O}(n \log n)$ sökum röðunar.
- Takið eftir að það er mjög auðvelt að gera litlar villur sem gera lausnin ranga.
- Til dæmis fær eftirfarandi lausn rétt í sýnidæmum en rangt á fyrsta huldudæminu.

# Planetaris

```c
#include <stdio.h>
#include <stdlib.h>
#define MAXN 100000
int cmp(const void* p1, const void* p2)
{
    int x = *(int*)p1, y = *(int*)p2;
    return (y <= x) - (x <= y);
}

int main()
{
    int i, n, a, e[MAXN];
    scanf("%d%d", &n, &a);
    for (i = 0; i < n; i++) scanf("%d", &e[i]);
    qsort(e, n, sizeof *e, cmp);
    for (i = 0; i < n; i++)
    {
        a -= e[i] + 1;
        if (a <= e[i] + 1) break;
    }
    printf("%d\n", i + 1);
    return 0;
}
```