Algoritmer og Datastrukturer

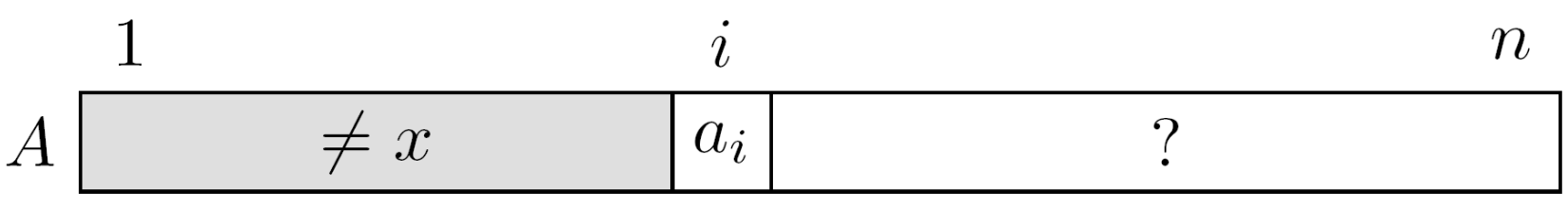
Binær og lineær søgning, logaritmer,
længste voksende delsekvens, Erdös Szekeres sætning

Søgning i usorteret liste

24	11	3	7	32	47	5	2	89	12

Find *x*

Visuel invariant



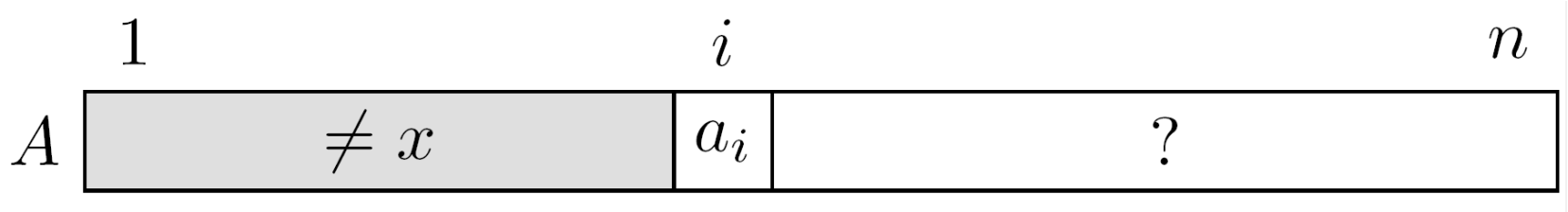
Formel invariant

$$1 \leq i \leq n+1 \ \wedge \ x_j \neq x \text{ for all } 1 \leq j < i$$

***n* sammenligninger**

Søgning i usorteret liste

Visuel invariant



Algorithm LinearSearch(A, x)
Input Array $A[1..n]$ and search key x
Output Index i where $A[i] = x$; -1 if $x \notin A$

```
1  i = 1
2  while i ≤ |A| do
3      if A[i] = x then
4          return i
5      i = i + 1
6  return −1
```

Søgning i sorteret liste

3	7	9	11	13	27	33	37	42	89

Find *x*

Visuel invariant

Formel invariant

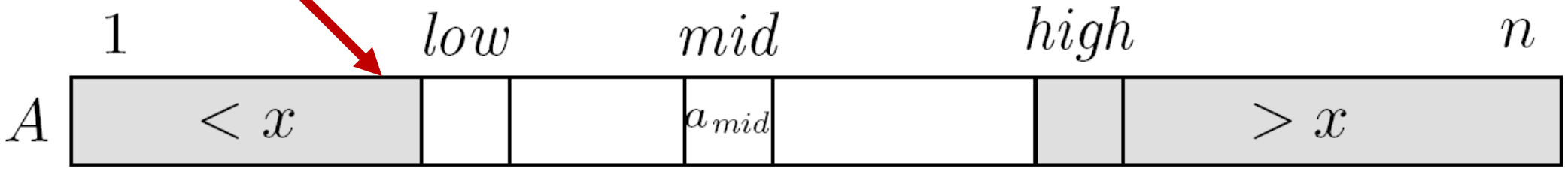
$$(1 \leq low \leq high \leq n+1) \ \wedge$$
$$(a_i < x \text{ for all } 1 \leq j < low) \ \wedge$$
$$(x < a_j \text{ for all } high \leq j \leq n)$$

Algorithm BINARYSEARCH(A, x)
Input Sorted array $A[1..n]$ and search key x
Output Index i where $A[i] = x$; -1 if $x \notin A$

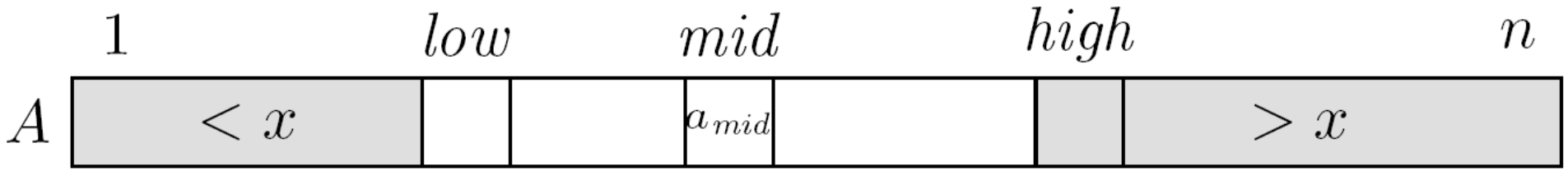
```
1   low = 1
2   high = n + 1
3   while low < high do
4       mid = ⌊(low + high)/2⌋
5       if x = A[mid] then
6           return mid
7       else if x < A[mid] then
8           high = mid
9       else if x > A[mid] then
10          low = mid + 1
11  return −1
```

Opgave
Hvad sker der når man runder op her?

Opgave
Modificer algoritmen til *low* peger her



Binær søgning – antal sammenligninger



Antal kandidater efter én sammenligning ≤ : $n \to \lfloor n/2 \rfloor$

$$n \to \lfloor n/2 \rfloor \to \lfloor \lfloor n/2 \rfloor /2 \rfloor \to \lfloor \lfloor \lfloor n/2 \rfloor /2 \rfloor /2 \rfloor \to \cdots \to 0$$

højst 1+ $\lfloor \log_2 n \rfloor$ sammenligninger

Søgning i sorteret liste – nedre grænse

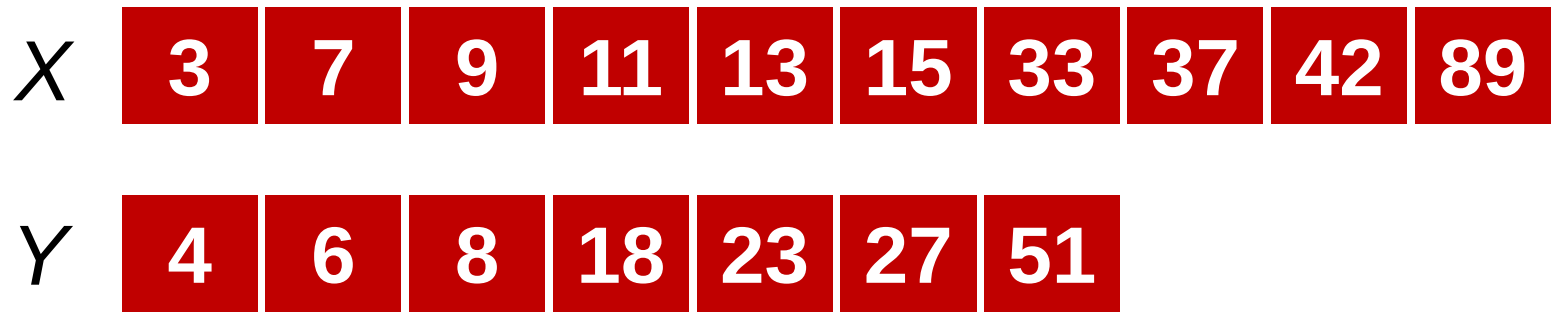
3	7	9	11	13	27	33	37	42	89

For enhver algoritme kan man svare således at kandidatmængden mindst er

$$n \to \lfloor n/2 \rfloor \to \lfloor \lfloor n/2 \rfloor /2 \rfloor \to \lfloor \lfloor \lfloor n/2 \rfloor /2 \rfloor /2 \rfloor \to \cdots \to 0$$

mindst $1+\lfloor \log_2 n \rfloor$ sammenligninger

Problem 1.9

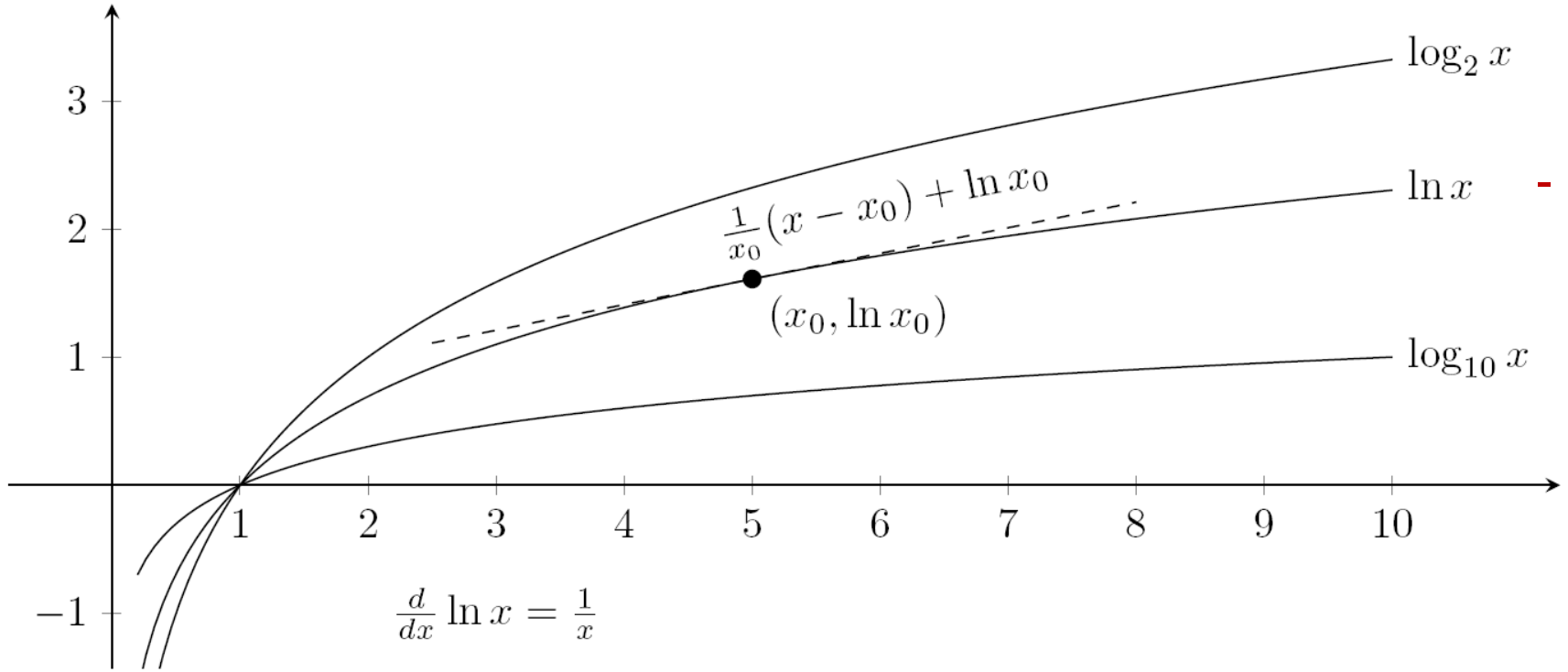


Givet X og Y sorteret, find det r-te mindste i $X \cup Y$

(det 9 mindste element er 15)

Logaritmer

Definition $y = \log_b x \Leftrightarrow b^y = x$



- naturlige logaritme

Logaritmer – regneregler

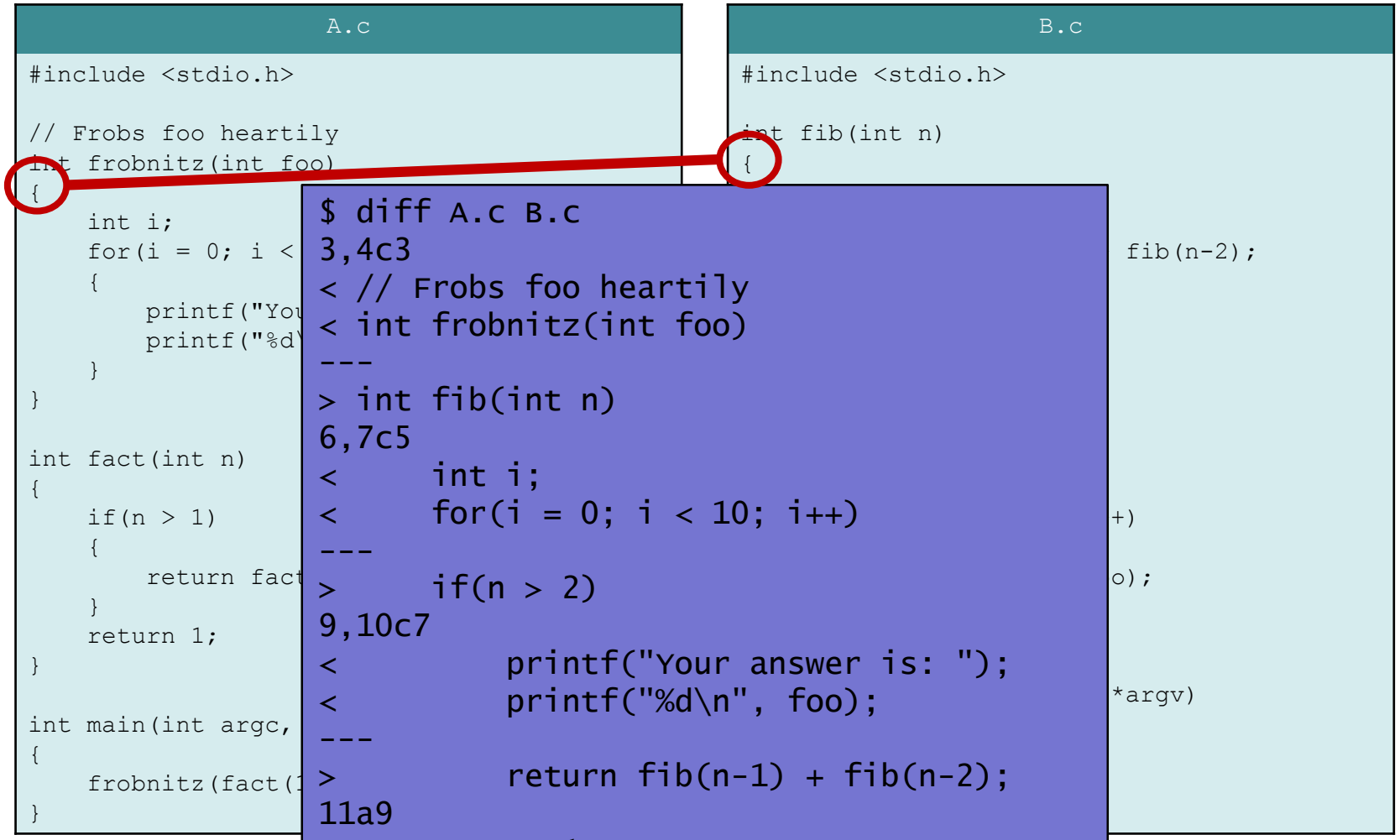
$$\begin{aligned}
\log_b(x \cdot y) &= \log_b x + \log_b y \\
\log_b(x/y) &= \log_b x - \log_b y \\
\log_b(x^p) &= p \cdot \log_b x \\
\log_b x &= \frac{\log_a x}{\log_a b} \\
\log_b(b^p) &= p \\
\frac{d}{dx} \ln x &= \frac{1}{x} \\
\ln y &= \int_1^y \frac{1}{x} dx \\
H_n - \ln n &\to \gamma \quad \text{for} \quad n \to \infty
\end{aligned}$$

Harmoniske tal $H_n = \sum_{i=1}^{n} \frac{1}{i}$

Euler-Mascheroni constant $\gamma = 0.577215664901\ldots$

Et algoritmisk eksempel (der anvender binær søgning)

Patience Diff
&
Længste Voksende Delsekvenser

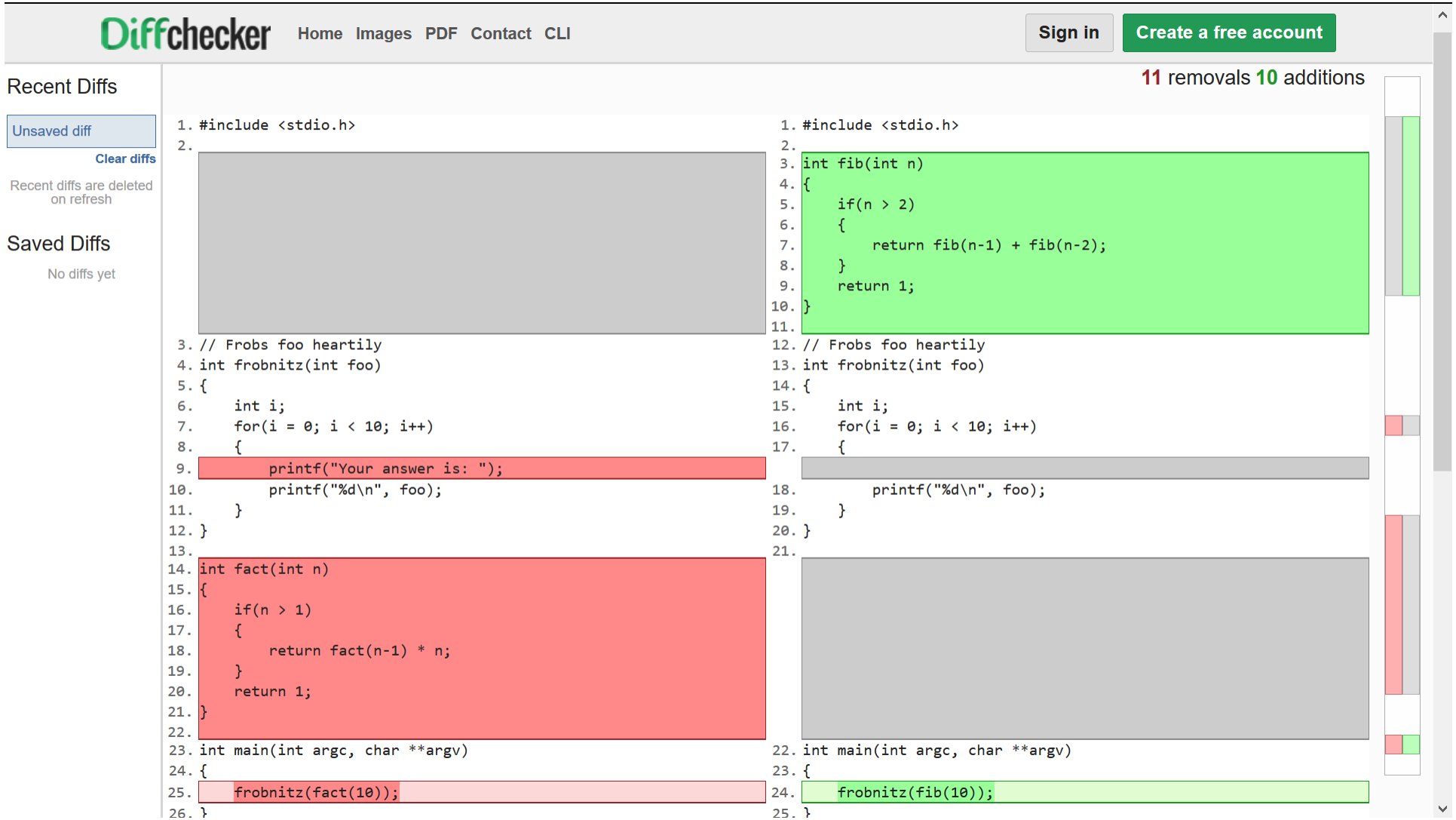

A.c
#include <stdio.h>
// Frobs foo heartily
int frobnitz(int foo)
{
int i;
for(i = 0; i <
{
printf("You
printf("%d\
}
}
int fact(int n)
{
if(n > 1)
{
return fact
}
return 1;
}
int main(int argc,
{
frobnitz(fact(1
}
B.c
#include <stdio.h>
int fib(int n)
{
fib(n-2);
+)
o);
*argv)
$ diff A.c B.c
3,4c3
< // Frobs foo heartily
< int frobnitz(int foo)

> int fib(int n)
6,7c5
< int i;
< for(i = 0; i < 10; i++)

> if(n > 2)
9,10c7
< printf("Your answer is: ");
< printf("%d\n", foo);

> return fib(n-1) + fib(n-2);
11a9
> return 1;
14c12,13
< int fact(int n)

> // Frobs foo heartily


Diffchecker
Home Images PDF Contact CLI
Sign in
Create a free account
Recent Diffs
Unsaved diff
Clear diffs
Recent diffs are deleted on refresh
Saved Diffs
No diffs yet
11 removals 10 additions
#include <stdio.h>
// Frobs foo heartily
int frobnitz(int foo)
{
int i;
for(i = 0; i < 10; i++)
{
printf("Your answer is: ");
printf("%d\n", foo);
}
}
int fact(int n)
{
if(n > 1)
{
return fact(n-1) * n;
}
return 1;
}
int main(int argc, char **argv)
{
frobnitz(fact(10));
}
#include <stdio.h>
int fib(int n)
{
if(n > 2)
{
return fib(n-1) + fib(n-2);
}
return 1;
}
// Frobs foo heartily
int frobnitz(int foo)
{
int i;
for(i = 0; i < 10; i++)
{
printf("%d\n", foo);
}
}
int main(int argc, char **argv)
{
frobnitz(fib(10));
}

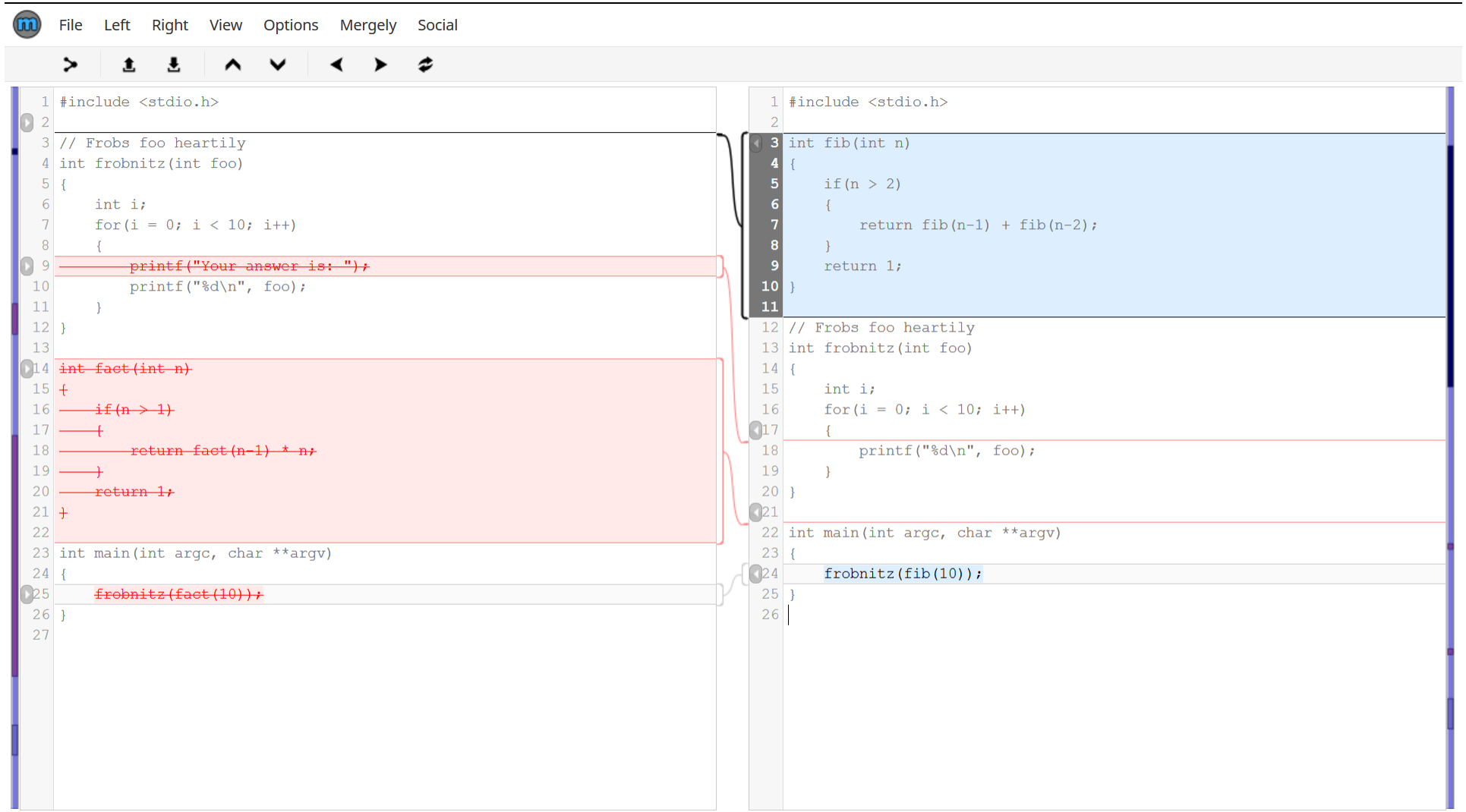
Text Compare!

Email this comparison

```
#include <stdio.h>

// Frobs foo heartily
int frobnitz(int foo)
{
    int i;
    for(i = 0; i < 10; i++)
    {
        printf("Your answer is: ");
        printf("%d\n", foo);
    }
}

int fact(int n)
{
    if(n > 1)
    {
        return fact(n-1) * n;
    }
    return 1;
}

int main(int argc, char **argv)
{
    frobnitz(fact(10));
}
```

```
#include <stdio.h>

int fib(int n)
{
    if(n > 2)
    {
        return fib(n-1) + fib(n-2);
    }
    return 1;
}

// Frobs foo heartily
int frobnitz(int foo)
{
    int i;
    for(i = 0; i < 10; i++)
    {

        printf("%d\n", foo);
    }
}

int main(int argc, char **argv)
{
    frobnitz(fib(10));
}
```

Edit texts ... | Switch texts | Compare! | Clear all

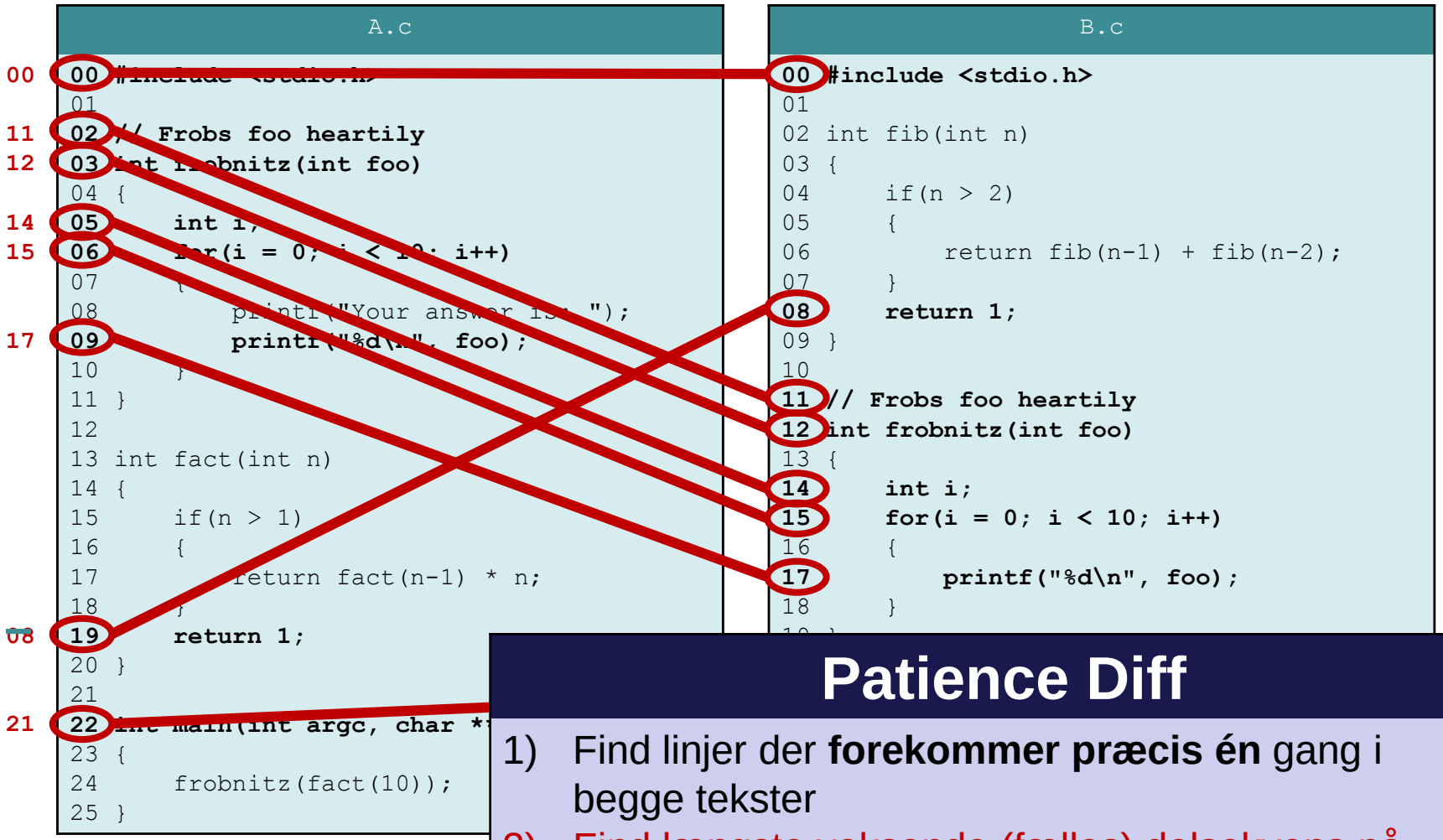

File
Left
Right
View
Options
Mergely
Social
#include <stdio.h>
// Frobs foo heartily
int frobnitz(int foo)
{
int i;
for(i = 0; i < 10; i++)
{
printf("Your answer is: ");
printf("%d\n", foo);
}
}
int fact(int n)
{
if(n > 1)
{
return fact(n-1) * n;
}
return 1;
}
int main(int argc, char **argv)
{
frobnitz(fact(10));
}
#include <stdio.h>
int fib(int n)
{
if(n > 2)
{
return fib(n-1) + fib(n-2);
}
return 1;
}
// Frobs foo heartily
int frobnitz(int foo)
{
int i;
for(i = 0; i < 10; i++)
{
printf("%d\n", foo);
}
}
int main(int argc, char **argv)
{
frobnitz(fib(10));
}

Patient Diff
(Bram Cohen, opfinder af BitTorrent)

- Forsøger at lave **læsbar og meningsfuldt output –** frem for mindst mulig

- Anvendes i Bazaar versionskontrolsystemet (bazaar-vcs.org)

Mindst mulig løsning findes senere i kurset vha. Dynamisk Programmering



Patience Diff

1) Find linjer der **forekommer præcis én** gang i begge tekster
2) Find længste voksende (fælles) delsekvens på disse
3) Gentag (rekursivt) på blokkende

Længste voksende delsekvens
(Problem 1.2)

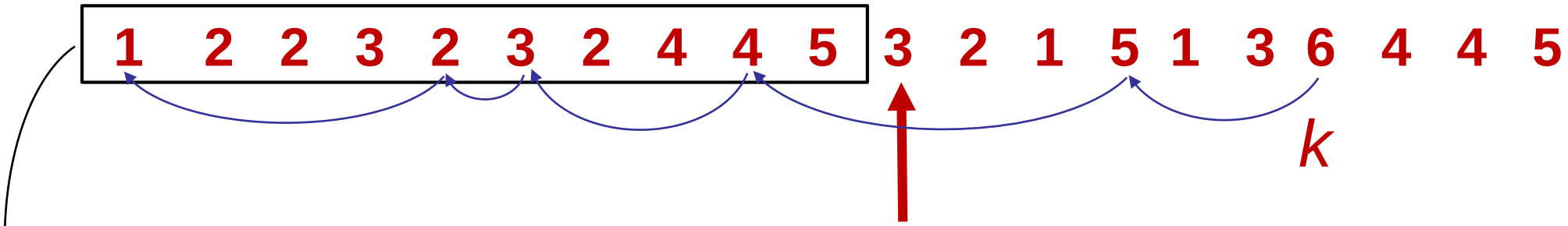
30 83 73 80 59 63 41 78 68 82 53 31 22 74 6 36 99 57 43 60

Opgave

Slet så få tal som muligt fra en liste af tal,
så de resterende tal står i voksende orden

Længste voksende delsekvens
(Problem 1.2)

30 83 73 80 59 63 41 78 68 82 53 31 22 74 6 36 99 57 43 60



længste voksende delsekvens der slutter på 53

Længde	Mindste sidste element
1	30
2	41
3	~~63~~ 53
4	68
5	82

= binær søgning

$\leq n \cdot (1 + \lfloor \log_2 k \rfloor)$ sammenligninger

Sætning (Erdős og Szekeres, 1935)

Enhver sekvens af n tal har en voksende eller aftagende delsekvens af længde mindst $\lceil \sqrt{n} \rceil$.

3 1 4 17 6 42 10 8 13 11 28 (voksende)

24 3 12 16 7 14 26 8 20 2 1 (aftagende)

Sætning (Erdős og Szekeres, 1935)

30 83 73 80 59 63 41 78 68 82 53 31 22 74 6 36 99 57 43 60

1 2 2 3 2 3 2 4 4 5 3 2 1 5 1 3 6 4 4 5

Længde	Mindste sidste element
1	30 22 6
2	83 73 59 41 31
3	80 63 53 36
4	78 68 57 43
5	74 60
6	99

Enten mange ($\geq \sqrt{n}$) rækker (lang voksende delsekvens) eller
mindst én lang ($\geq \sqrt{n}$) række (lang aftagende delsekvens)