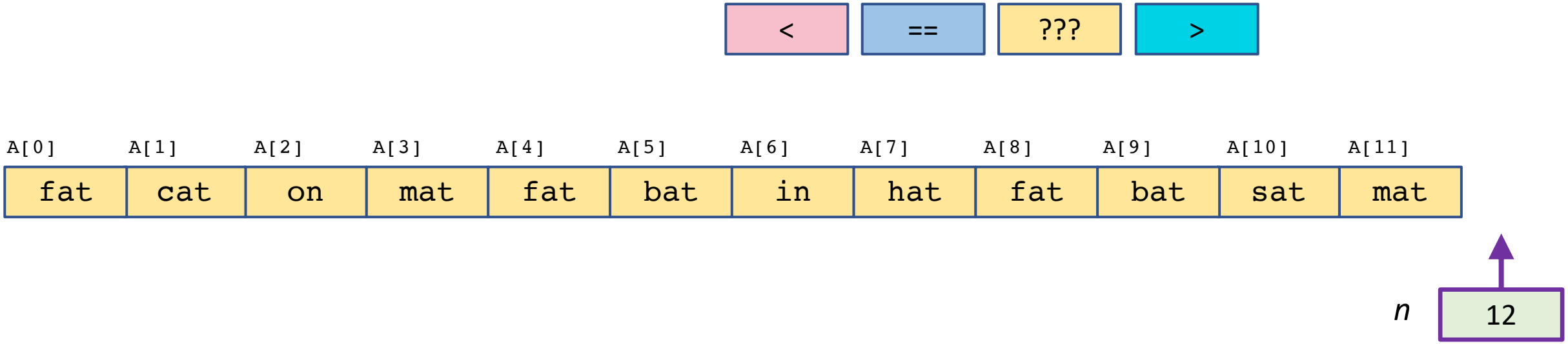
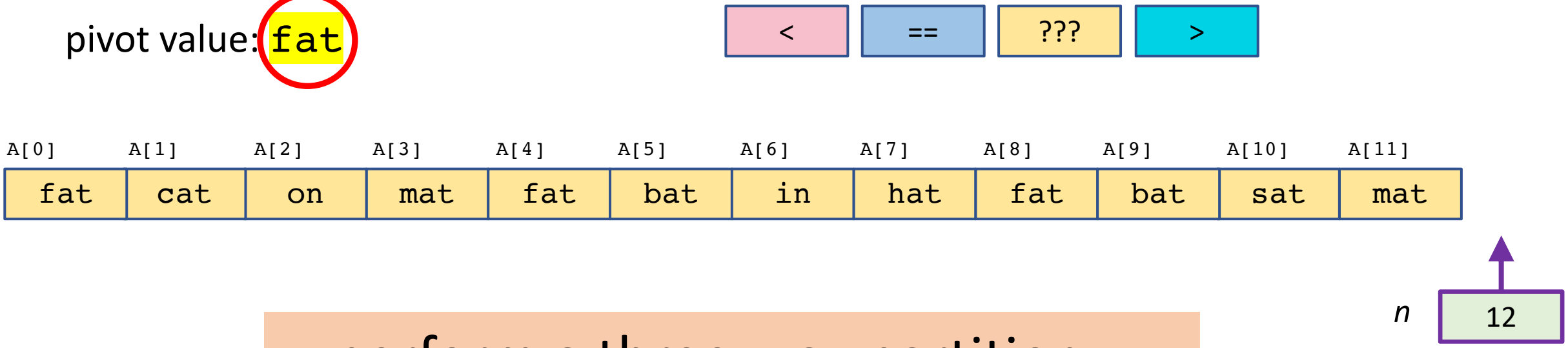
comp10002/comp20005
program quicksort.c

Support material for Chapter 12 of
Programming, Problem Solving, and Abstraction with C,
by Alistair Moffat

(c) 2023, The University of Melbourne
Prepared by Alistair Moffat, ammoffat@unimelb.edu.au


<
==
???
>
A[0]
A[1]
A[2]
A[3]
A[4]
A[5]
A[6]
A[7]
A[8]
A[9]
A[10]
A[11]
fat
cat
on
mat
fat
bat
in
hat
fat
bat
sat
mat
n
12

pivot value: `fat`



perform a three-way partition...

pivot value: fat

<	==	???	>

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]	A[11]
fat	cat	on	mat	fat	bat	in	hat	fat	bat	sat	mat

n 12

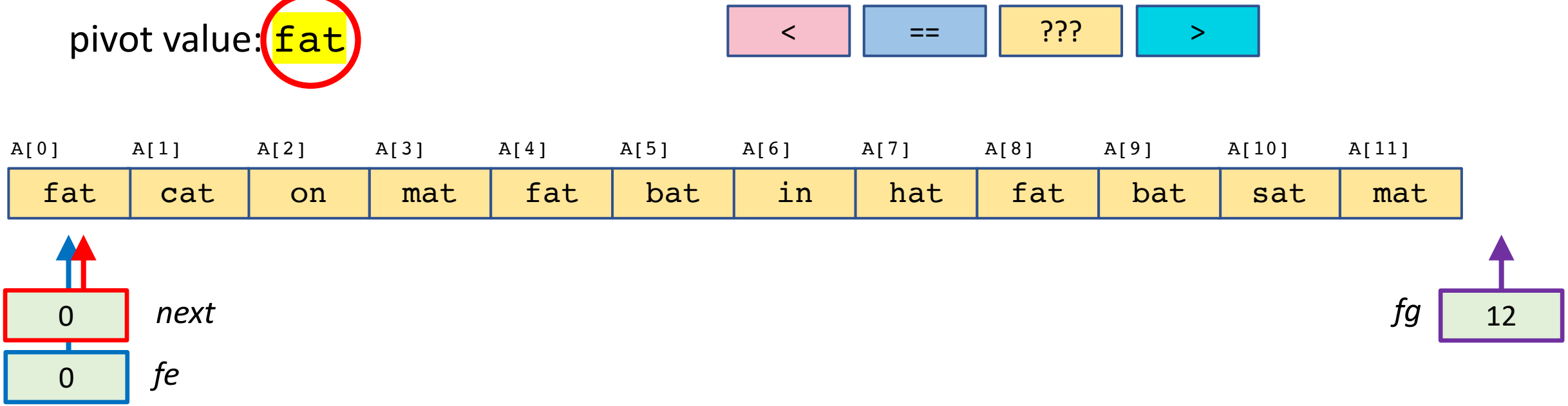
perform a three-way partition...

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]	A[11]
cat	bat	bat	fat	fat	fat	hat	in	mat	sat	mat	on

3 *fe*

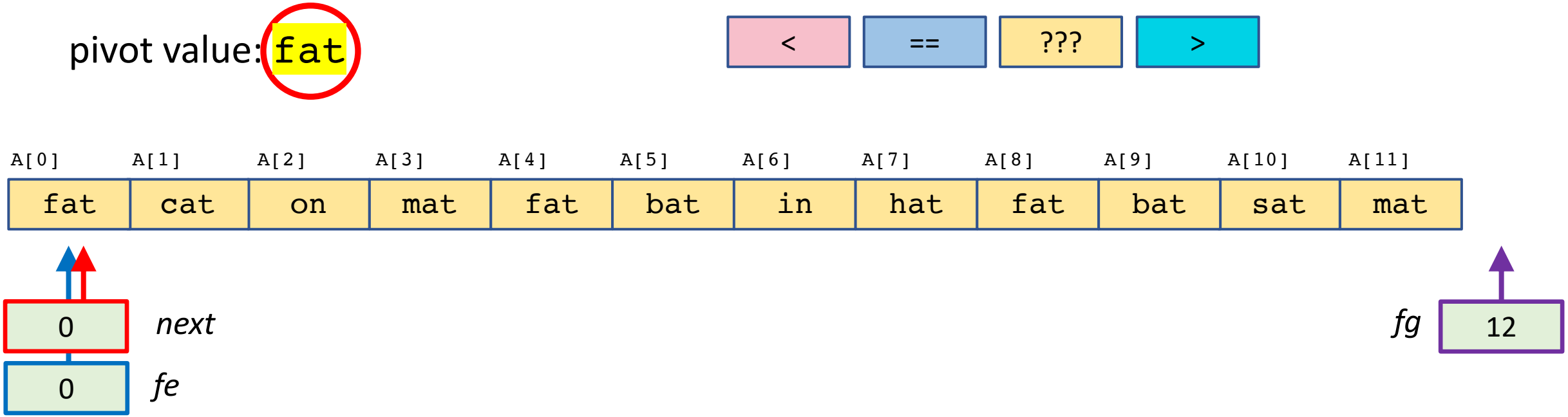
6 *fg*

pivot value: `fat`



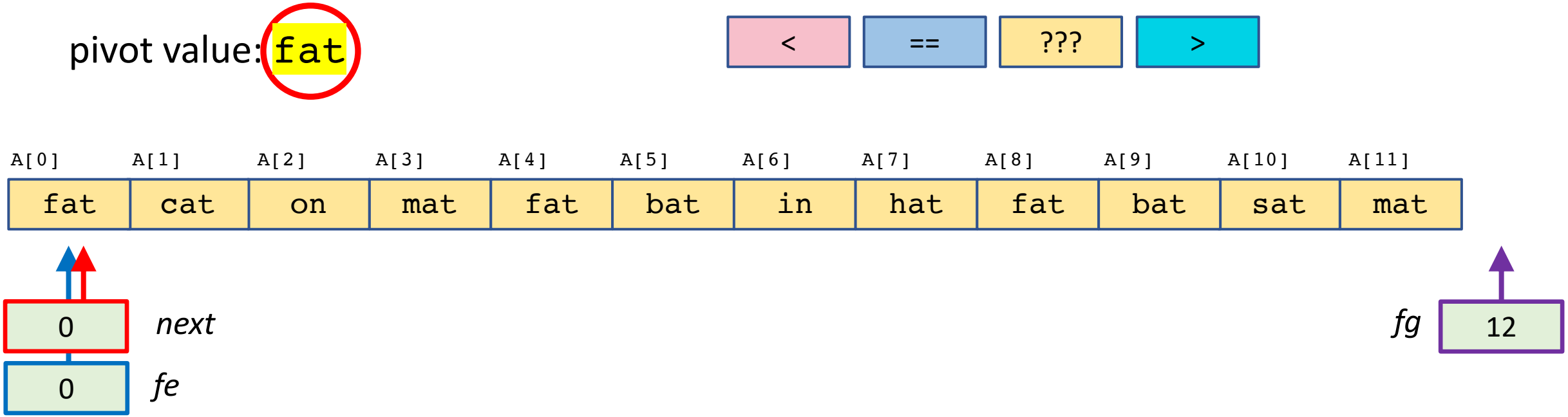
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`




```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`




```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

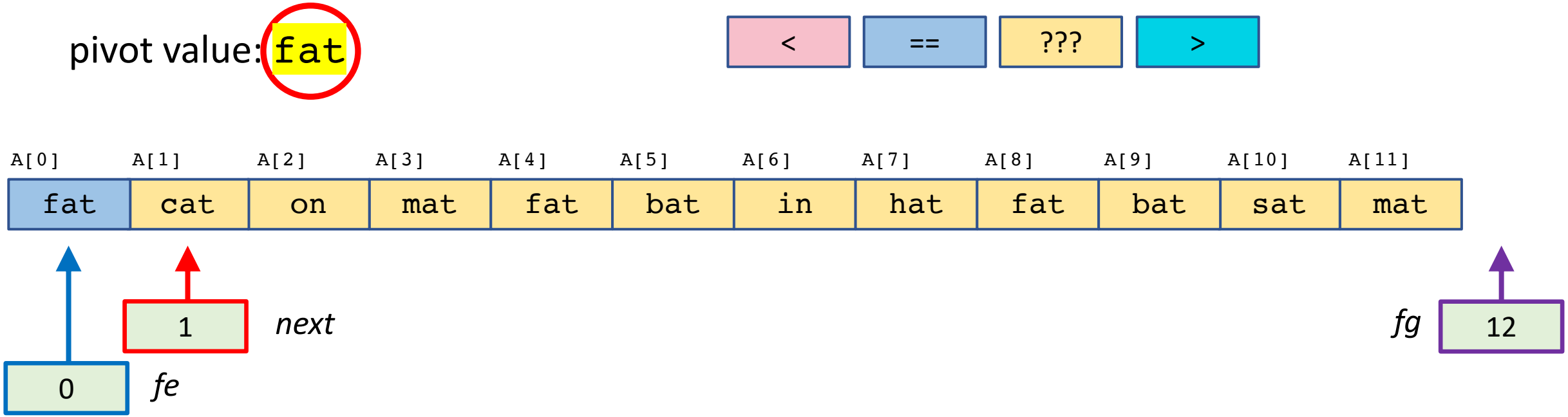
pivot value: `fat`

<	==	???	>

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]	A[11]
fat	cat	on	mat	fat	bat	in	hat	fat	bat	sat	mat

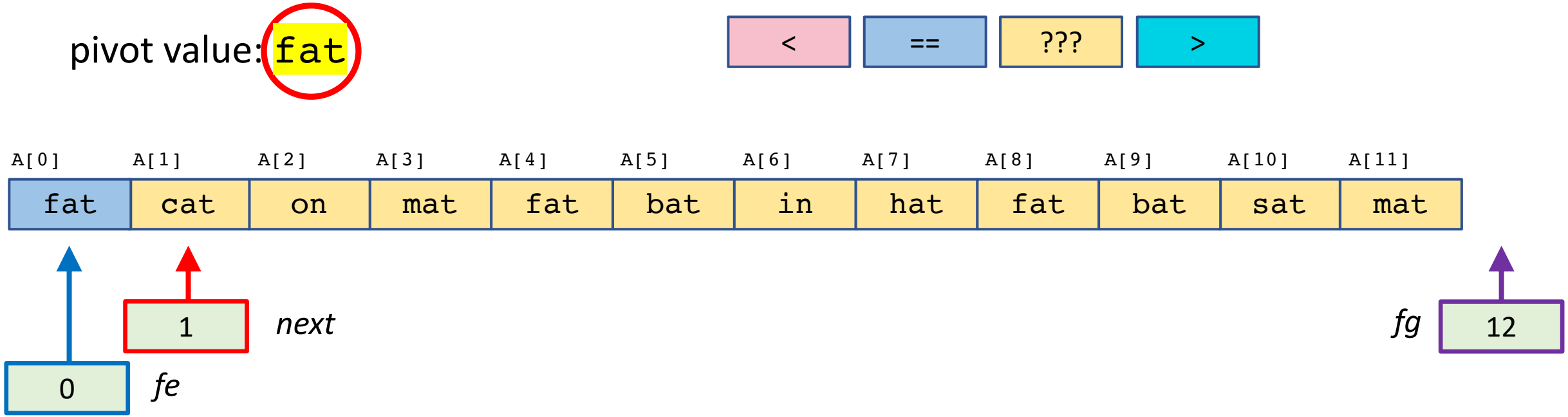
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



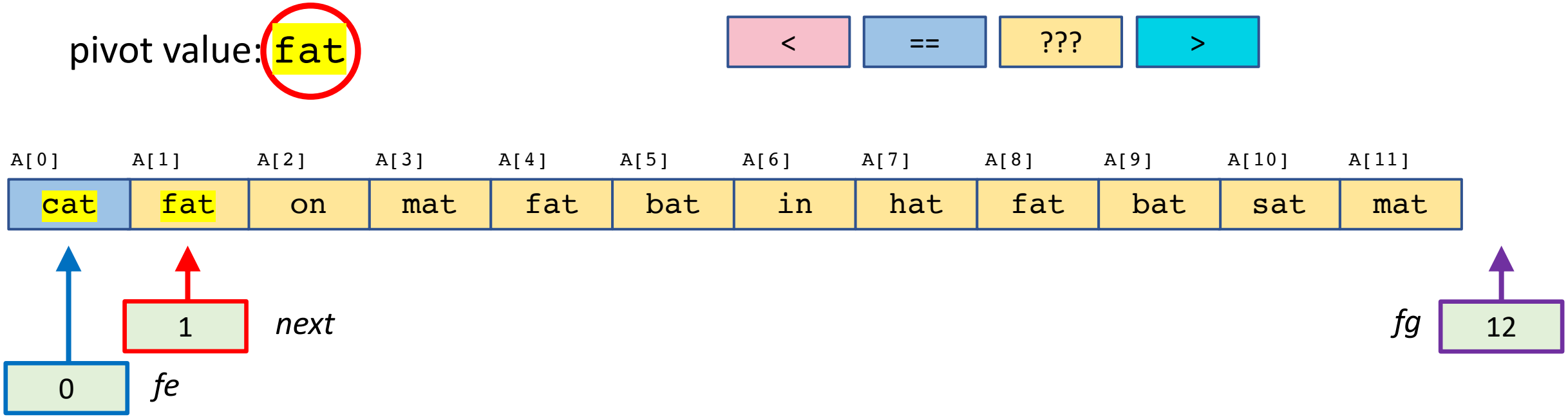

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



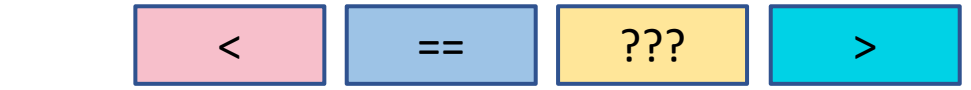

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

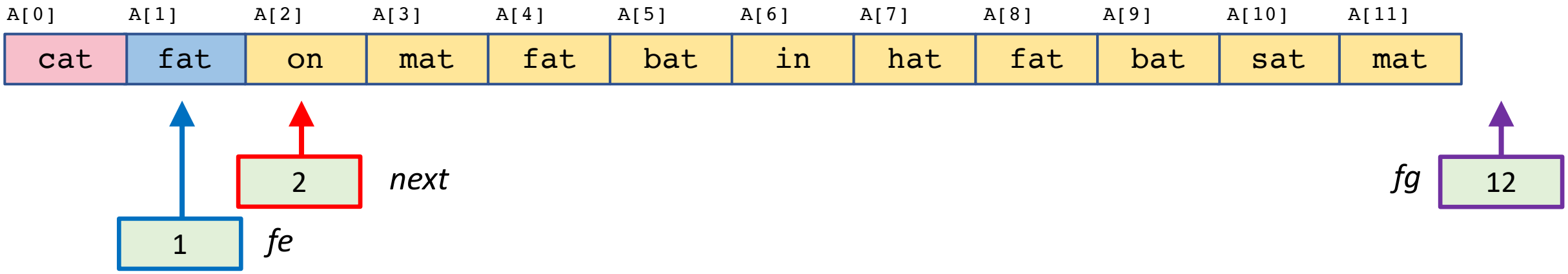
pivot value: fat



next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

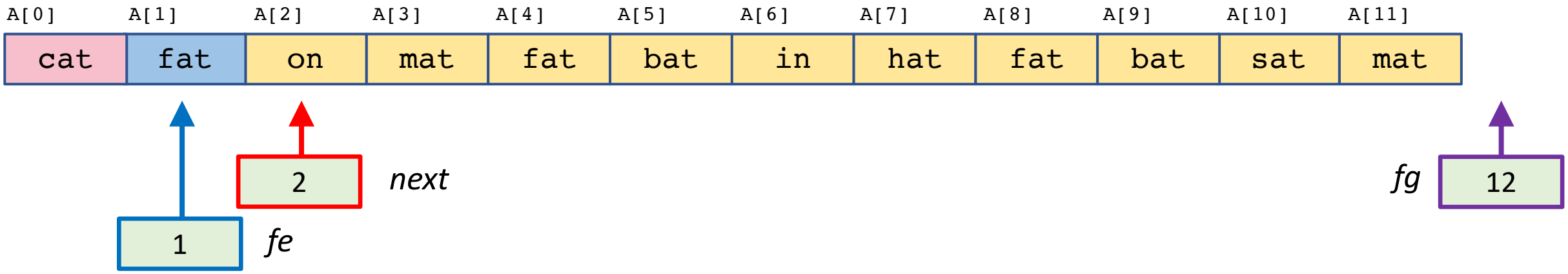
pivot value: `fat`





next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

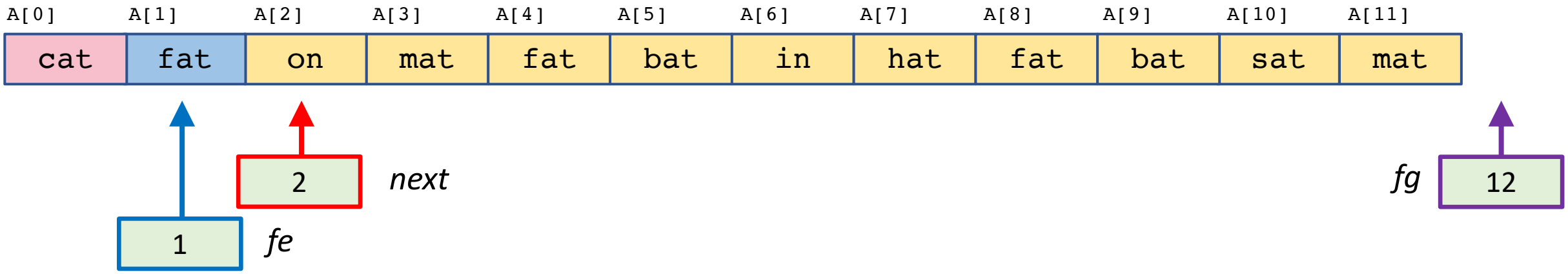
pivot value: fat



next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

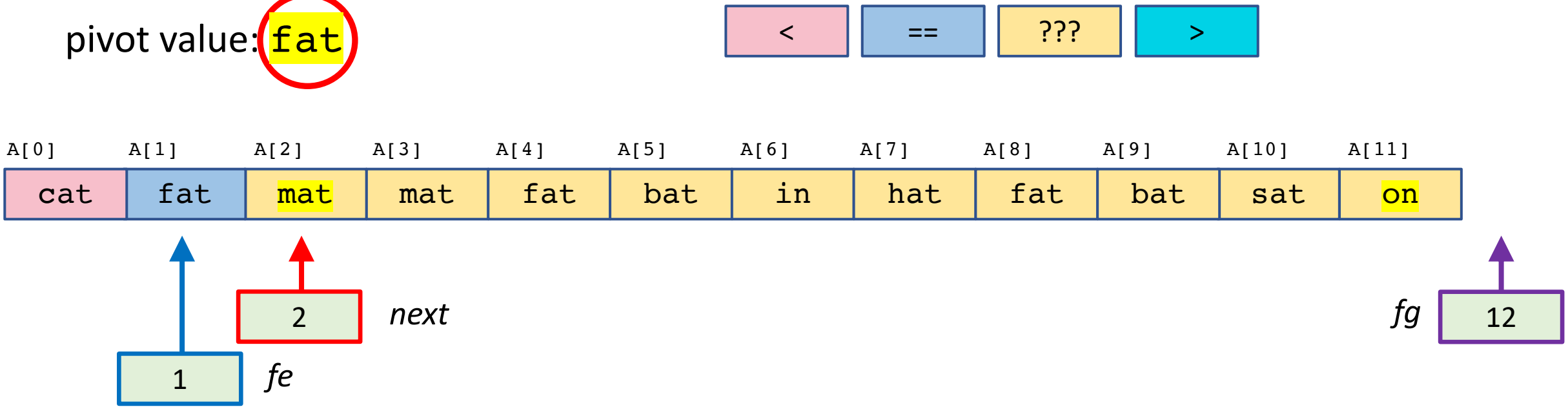
pivot value: `fat`

<	==	???	>



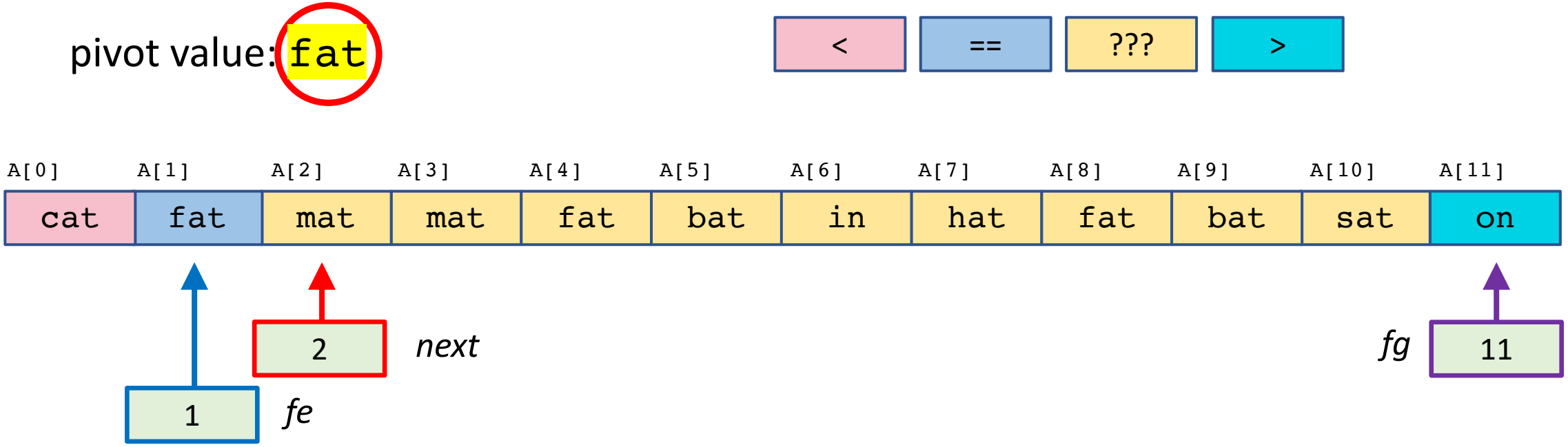

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: fat



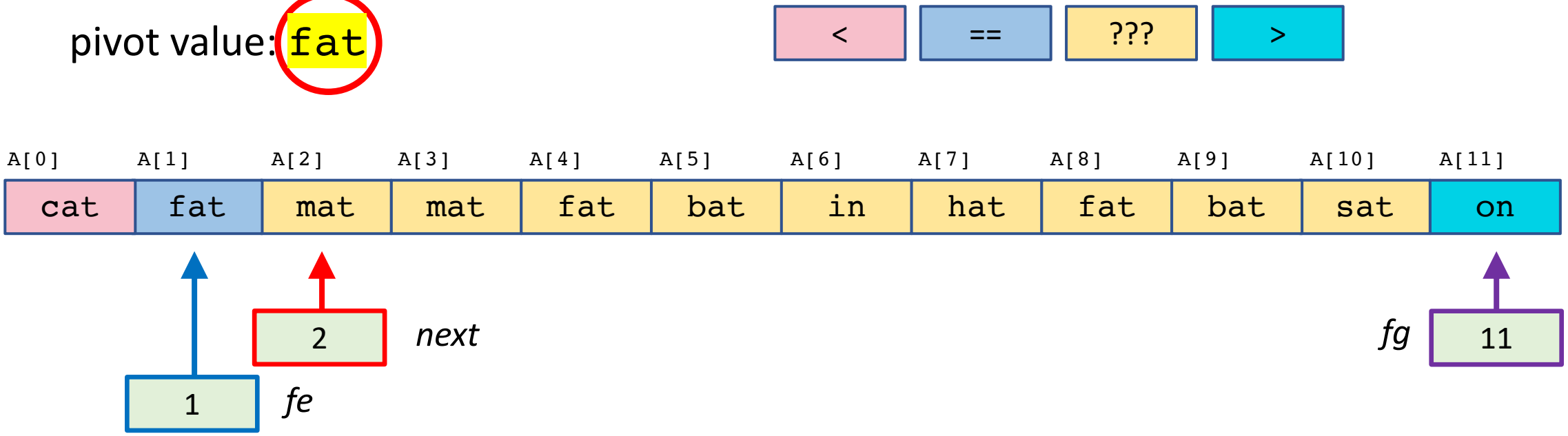
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



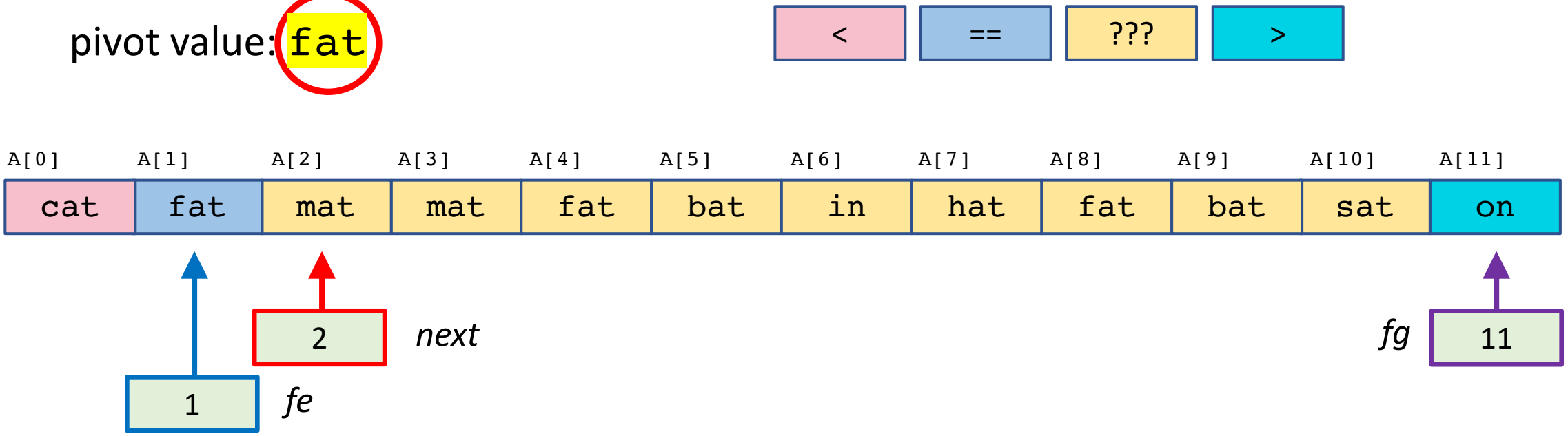
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



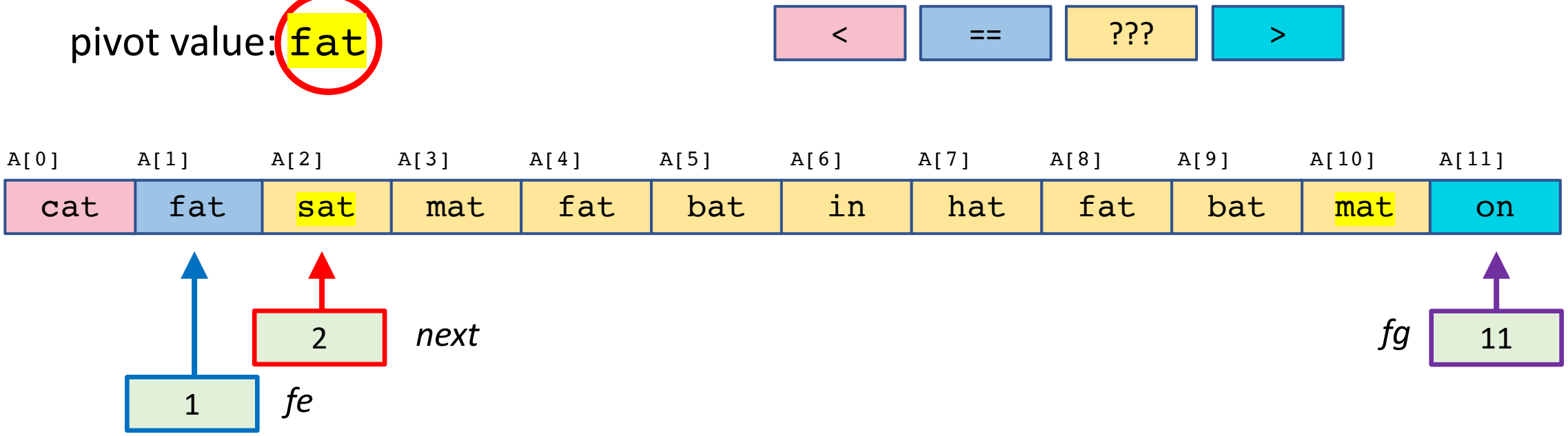

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



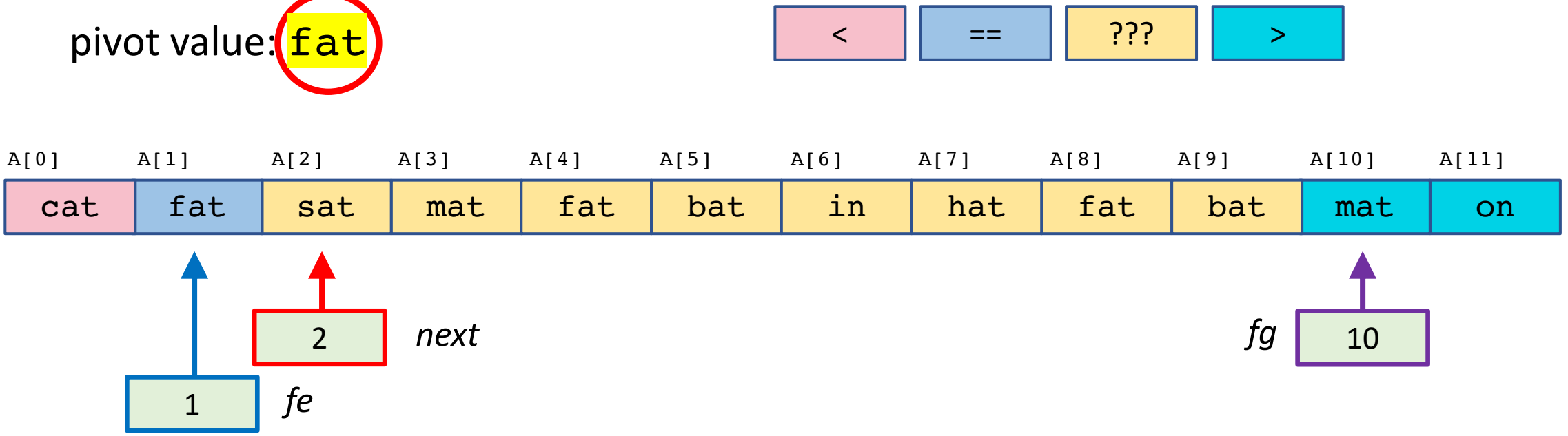
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: fat



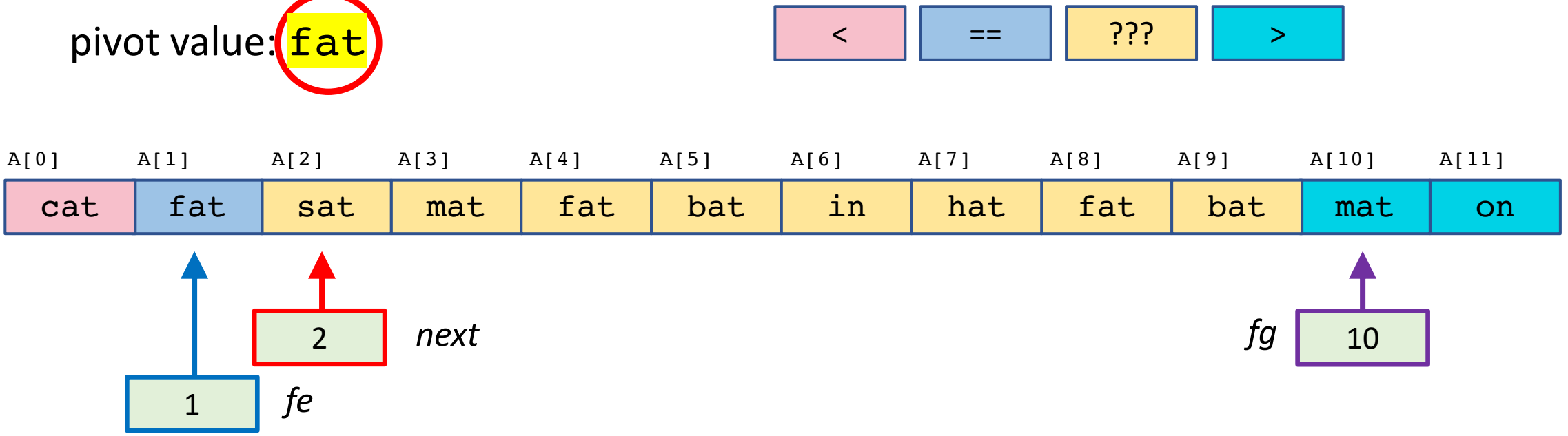
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: fat




```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

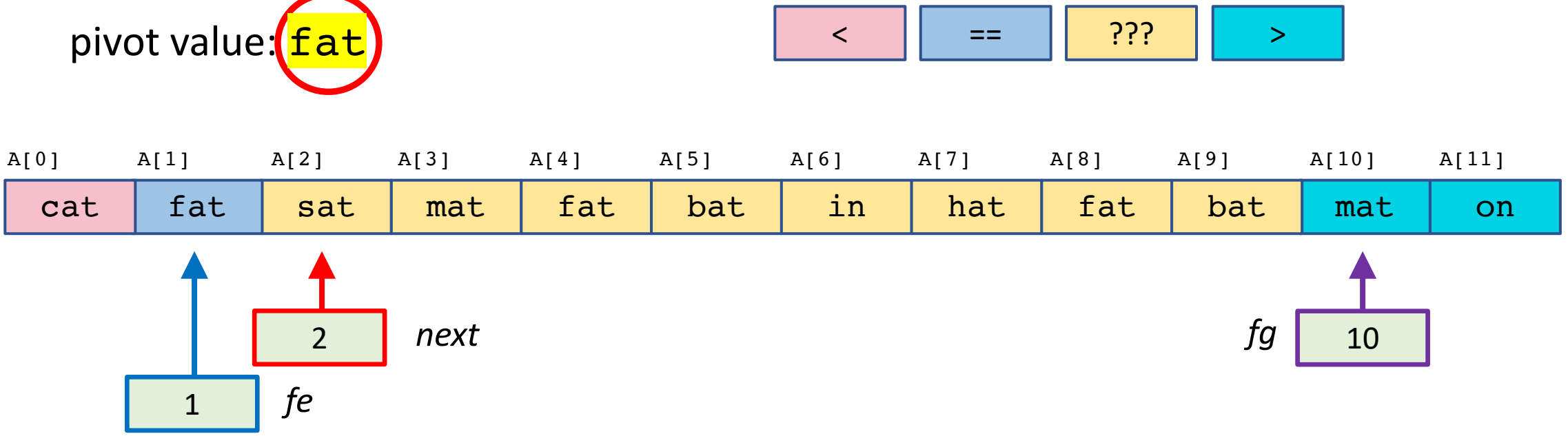
pivot value: fat



< == ??? >

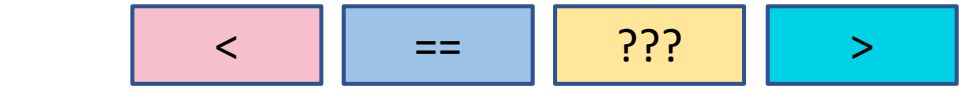
```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

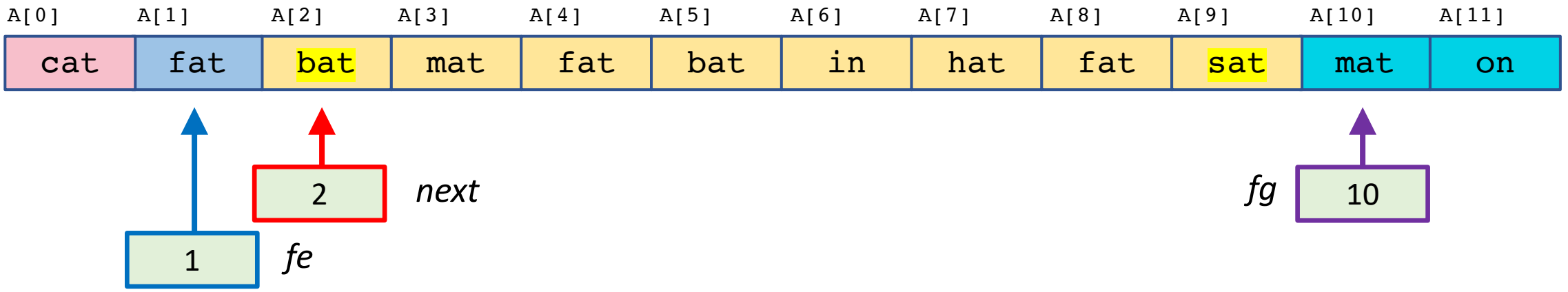
pivot value: `fat`




```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

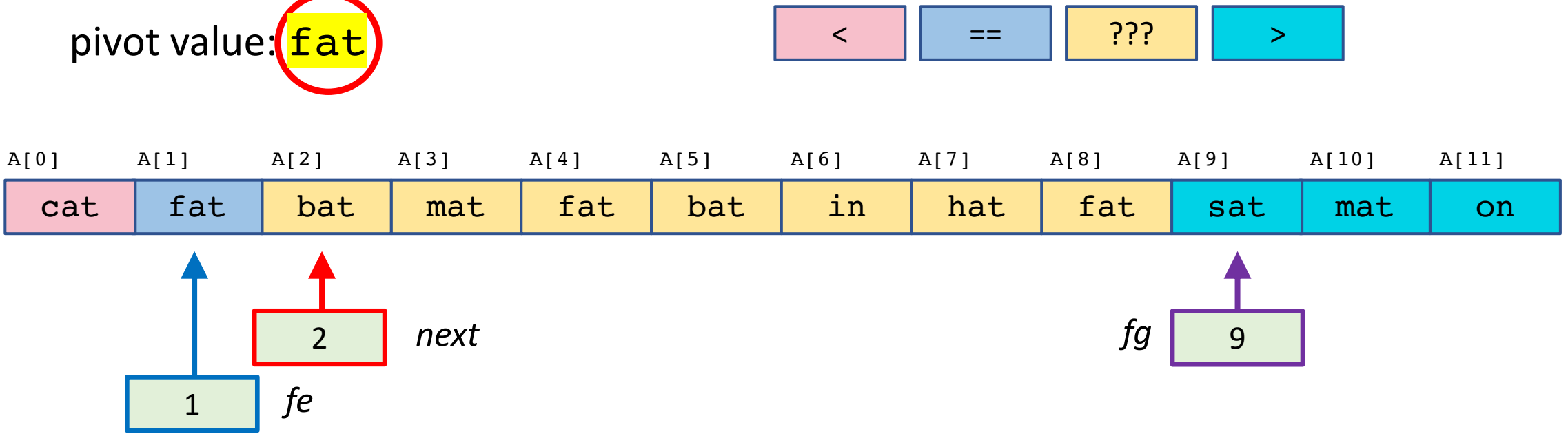
pivot value: fat





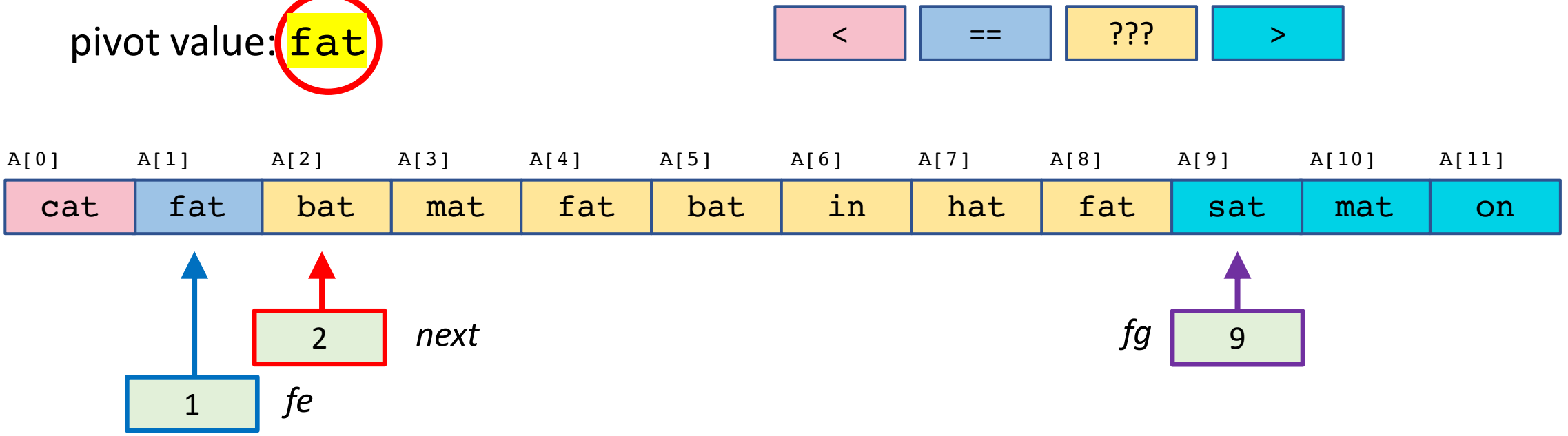
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



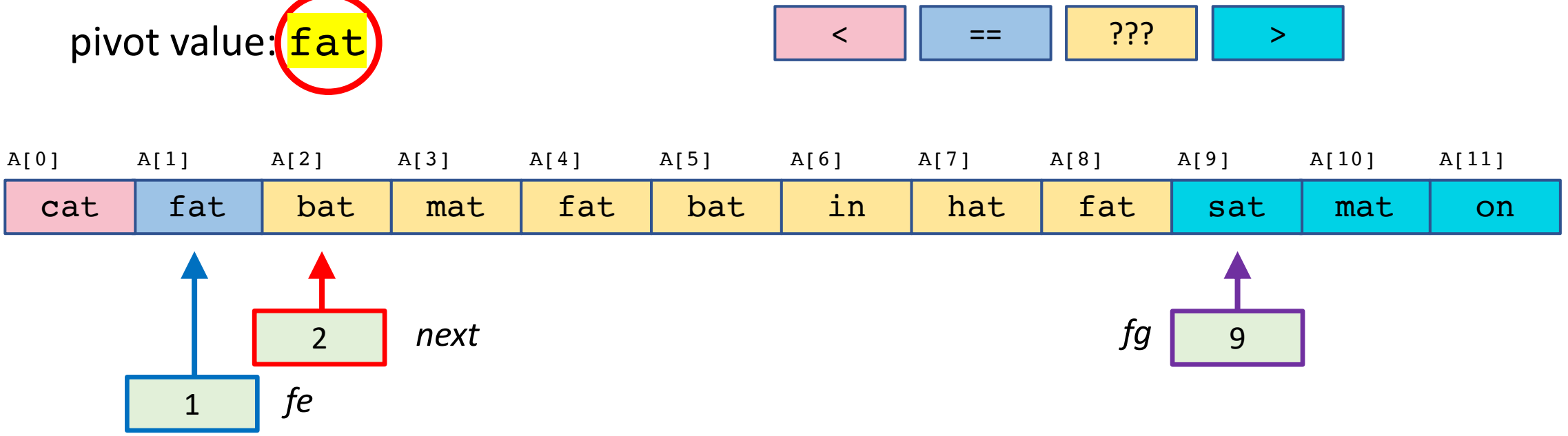

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



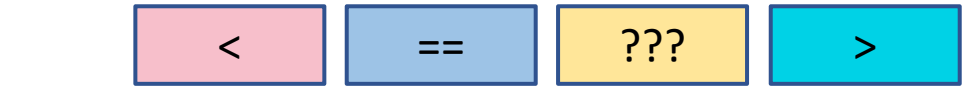
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

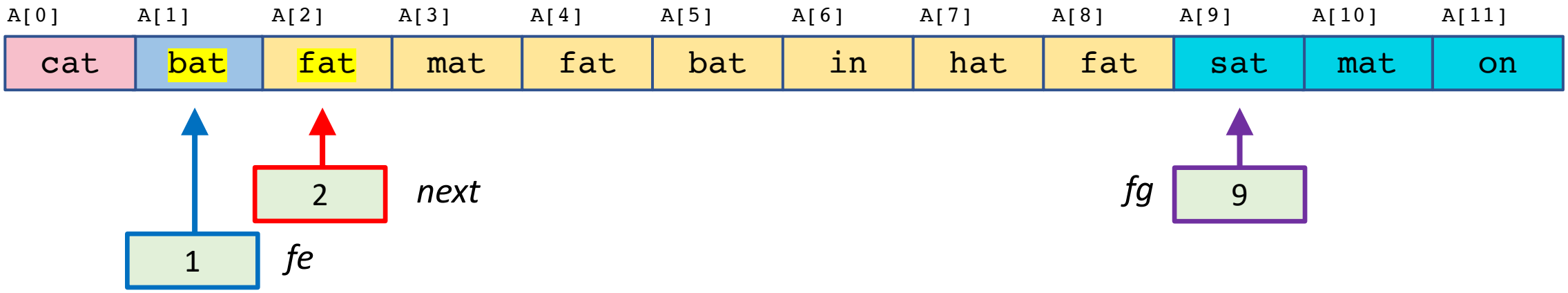
pivot value: `fat`




```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

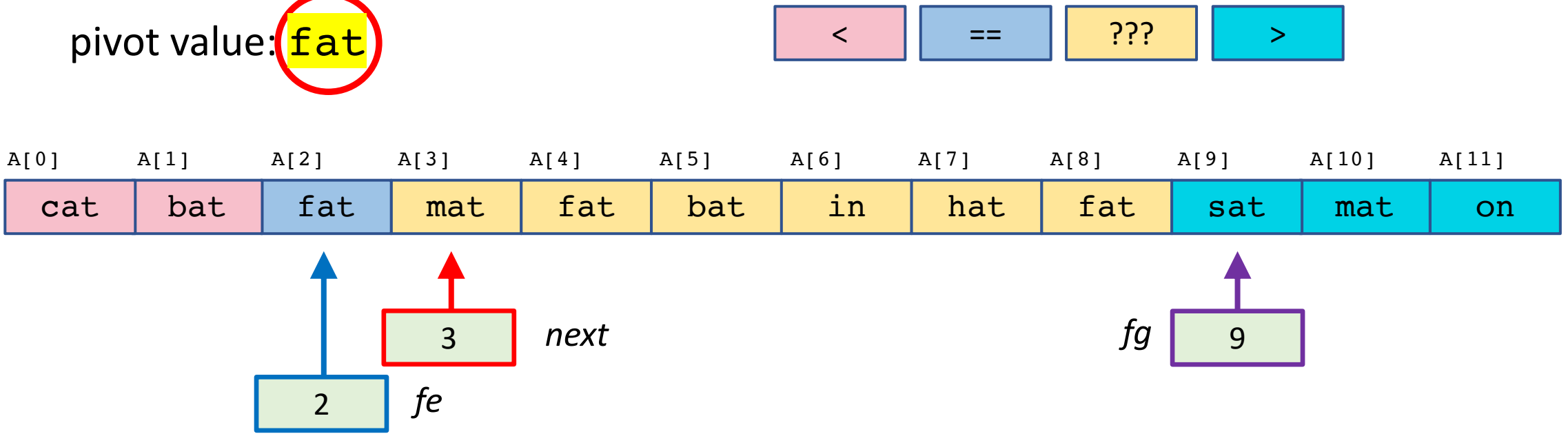
pivot value: fat





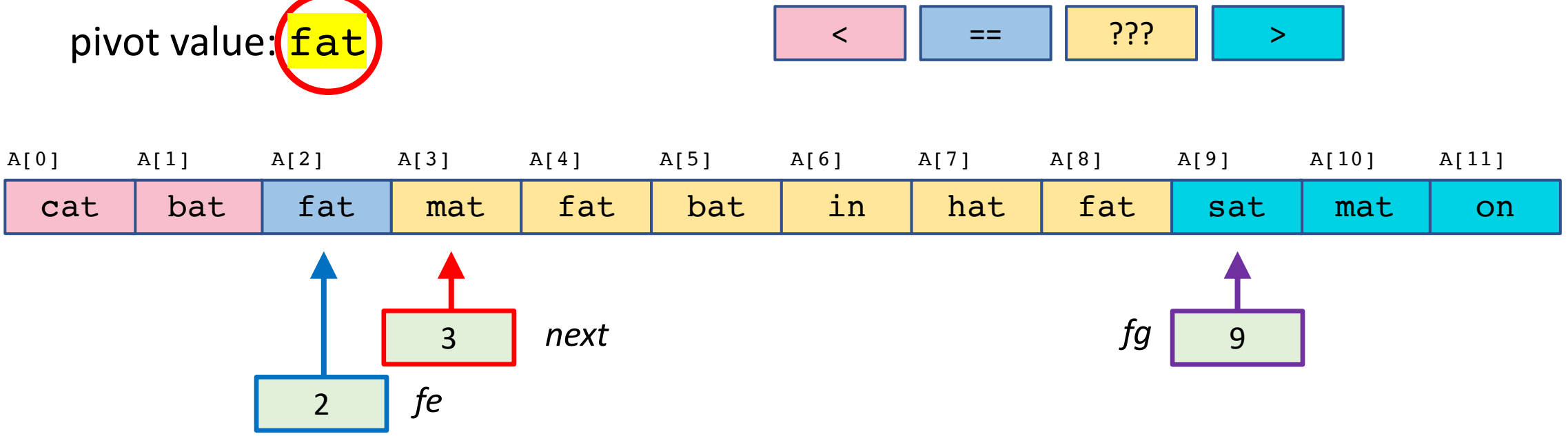
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



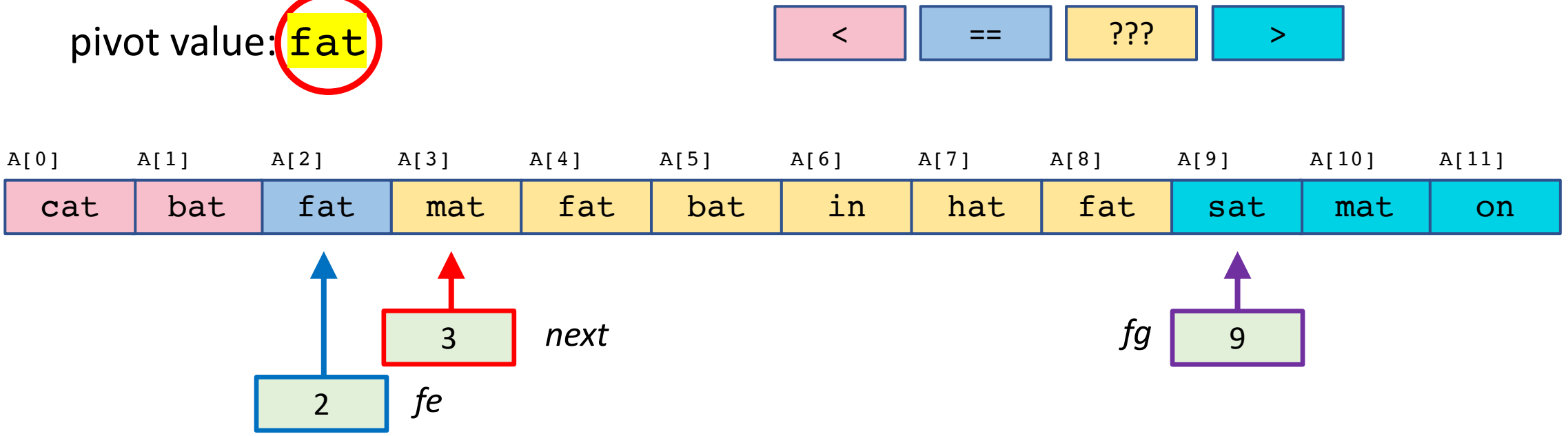
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



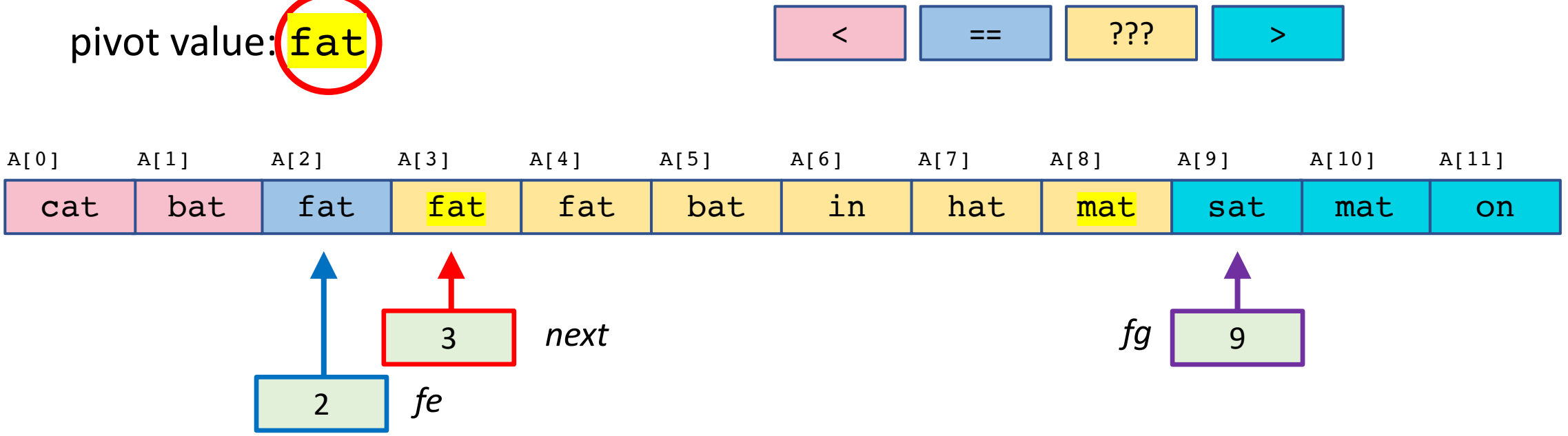

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



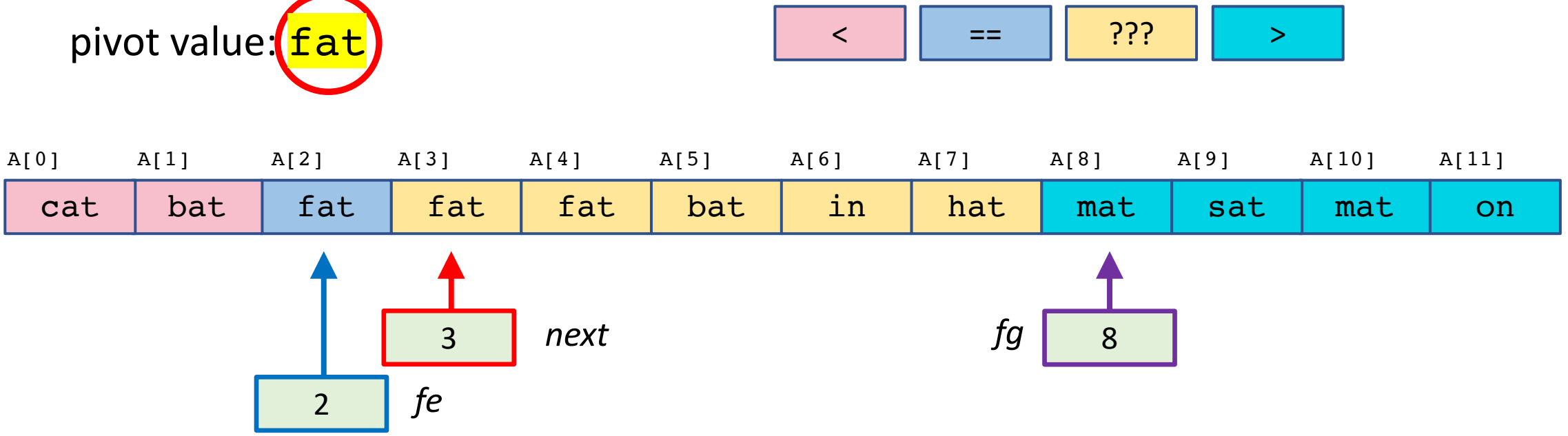
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: fat



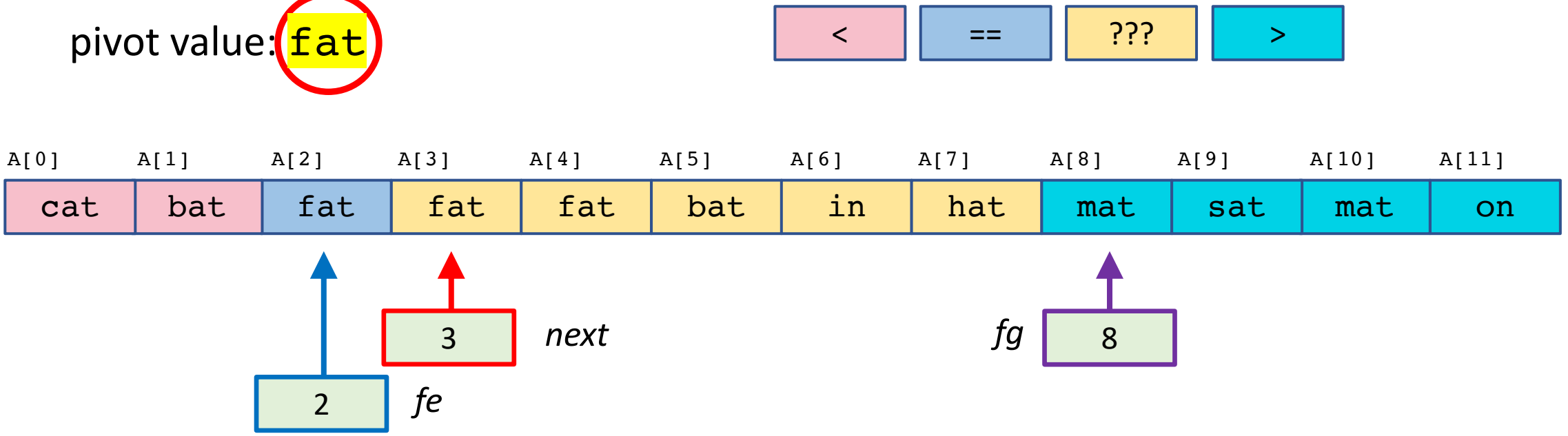

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



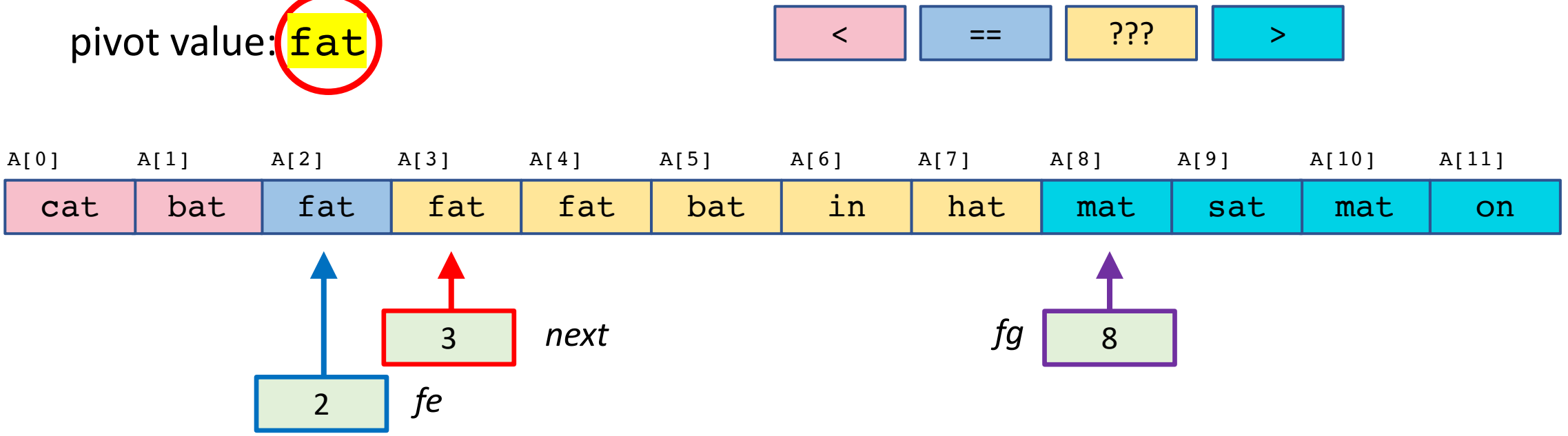

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



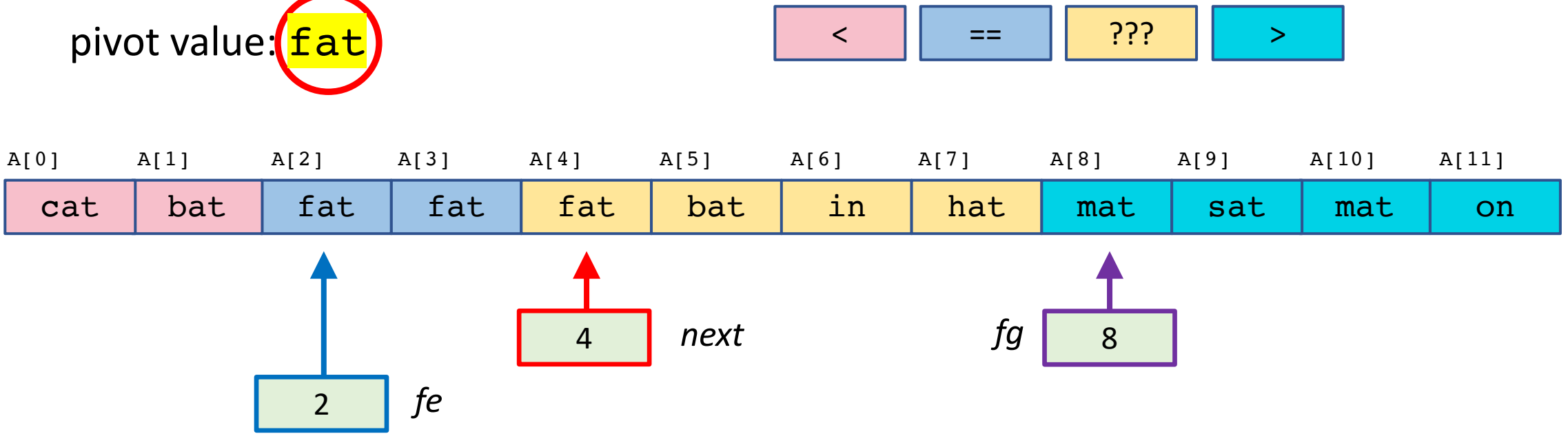

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



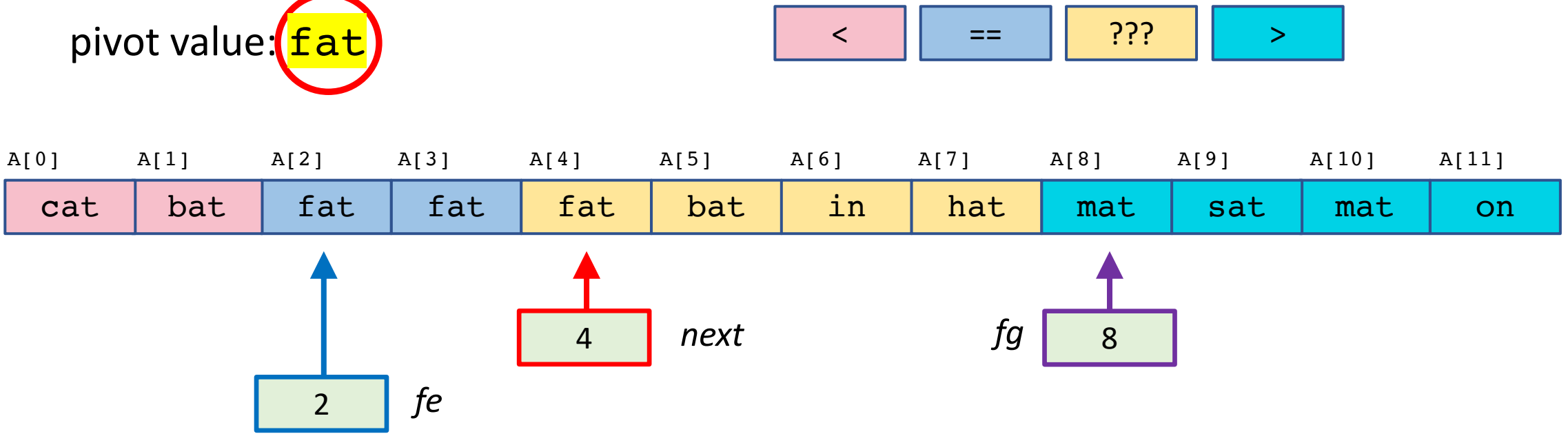

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



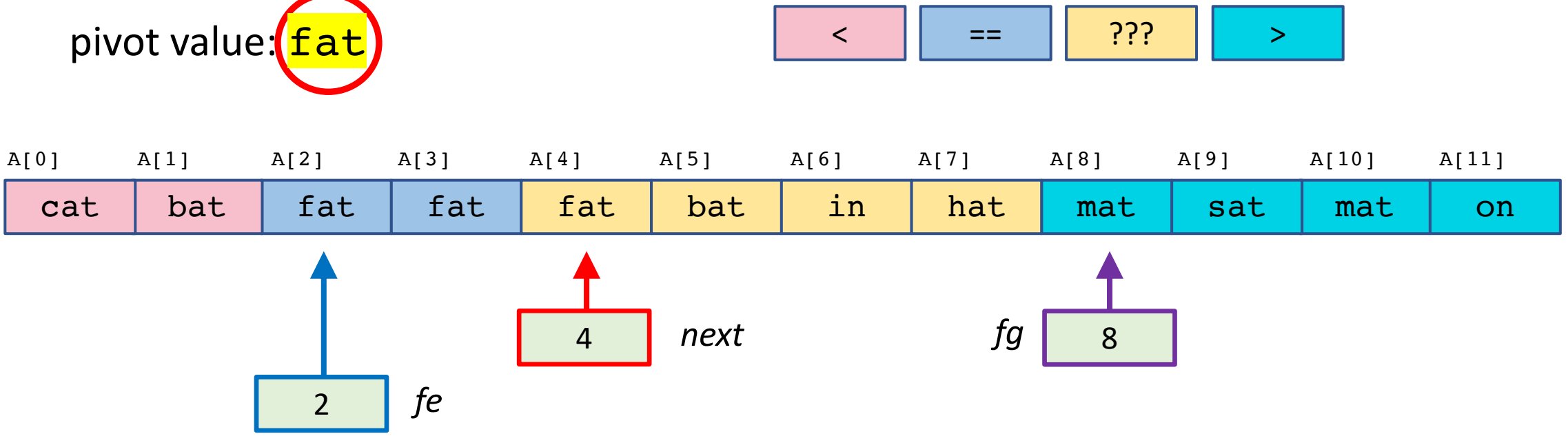

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



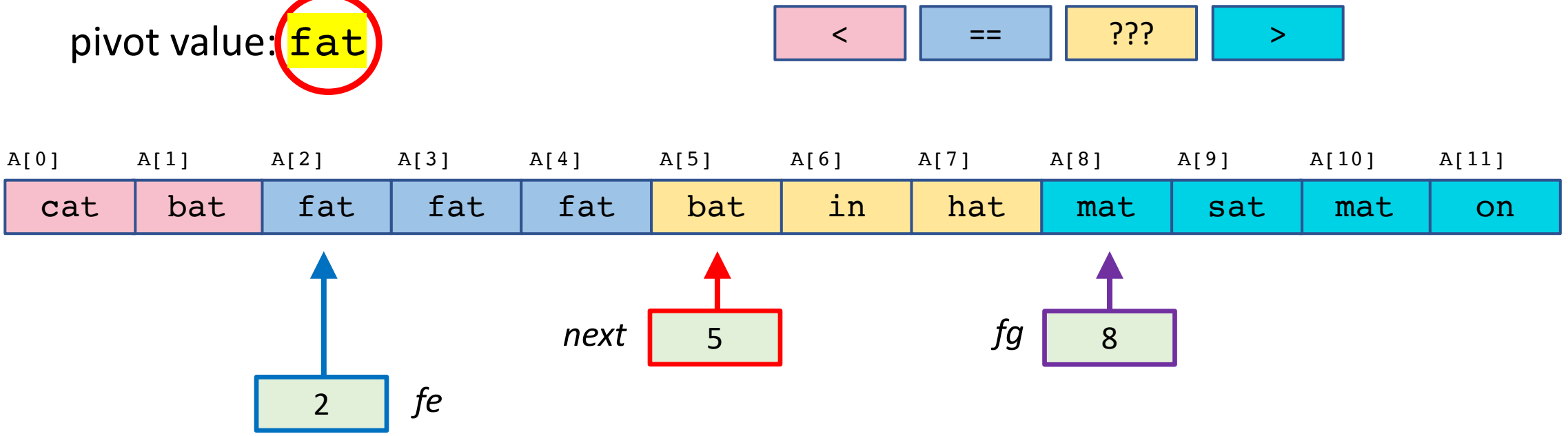

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



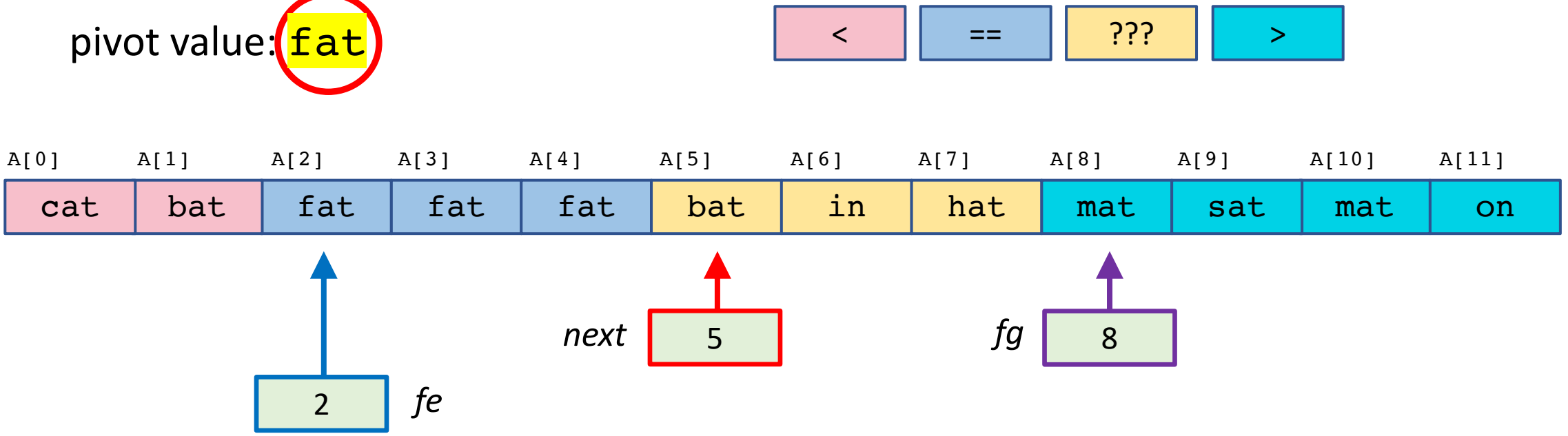

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

pivot value: `fat`



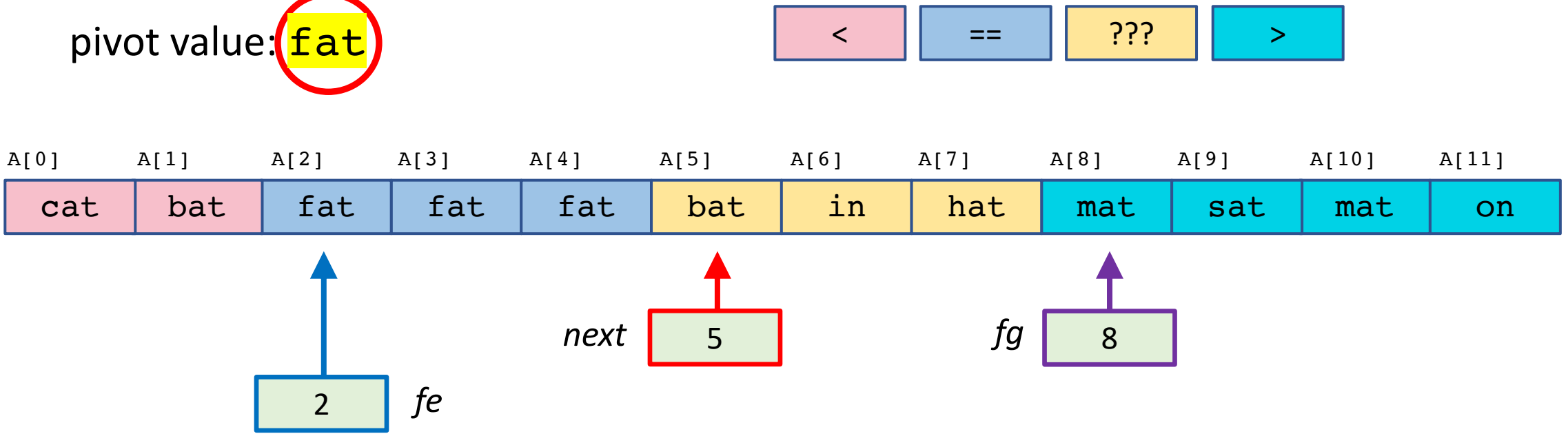
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



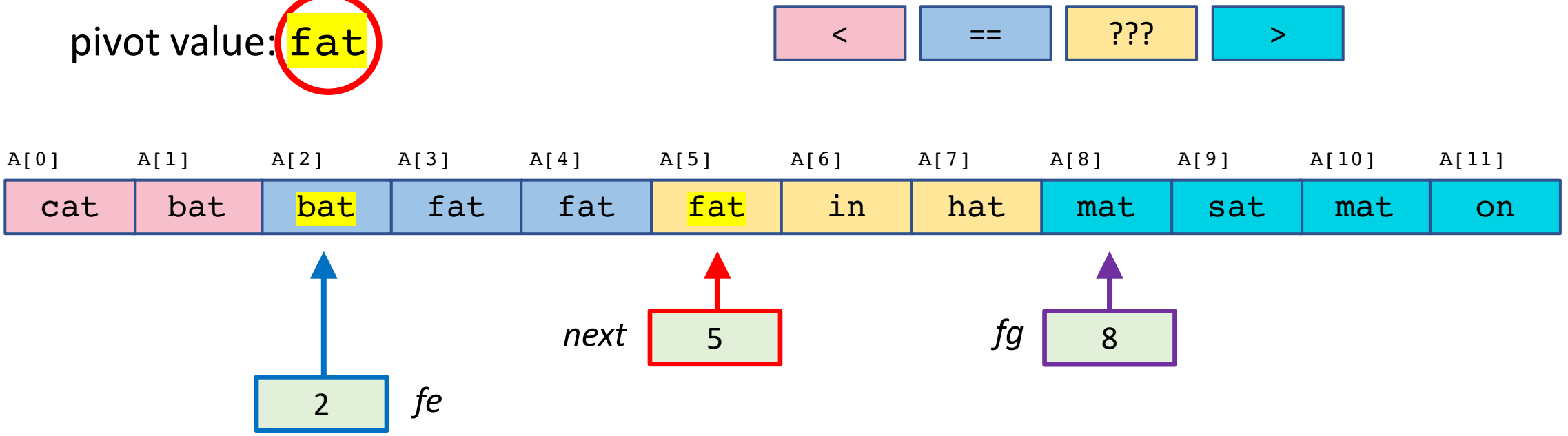
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



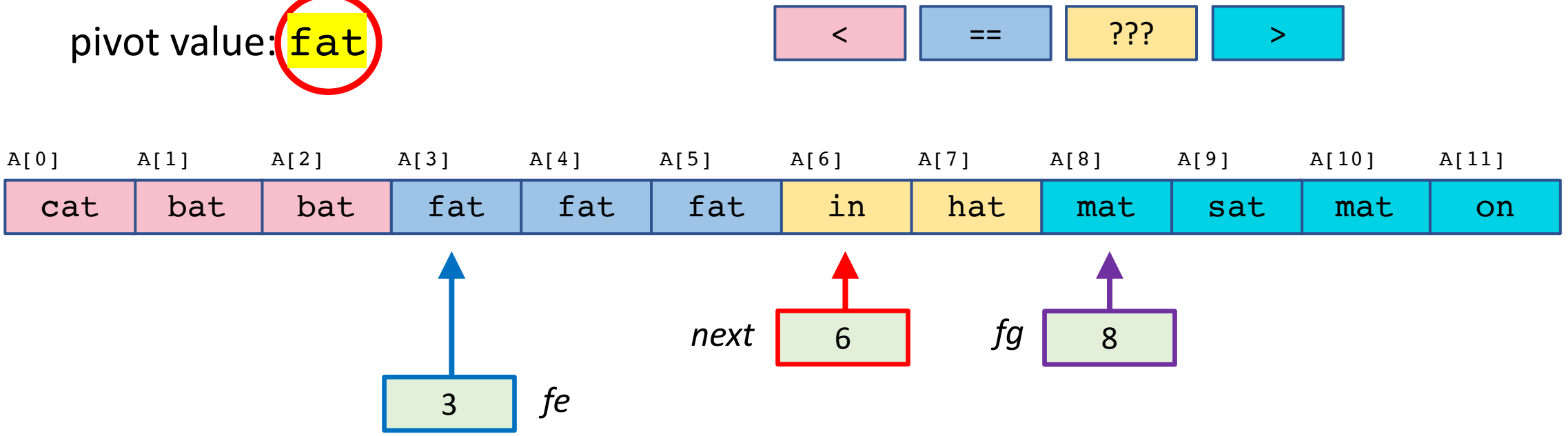
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



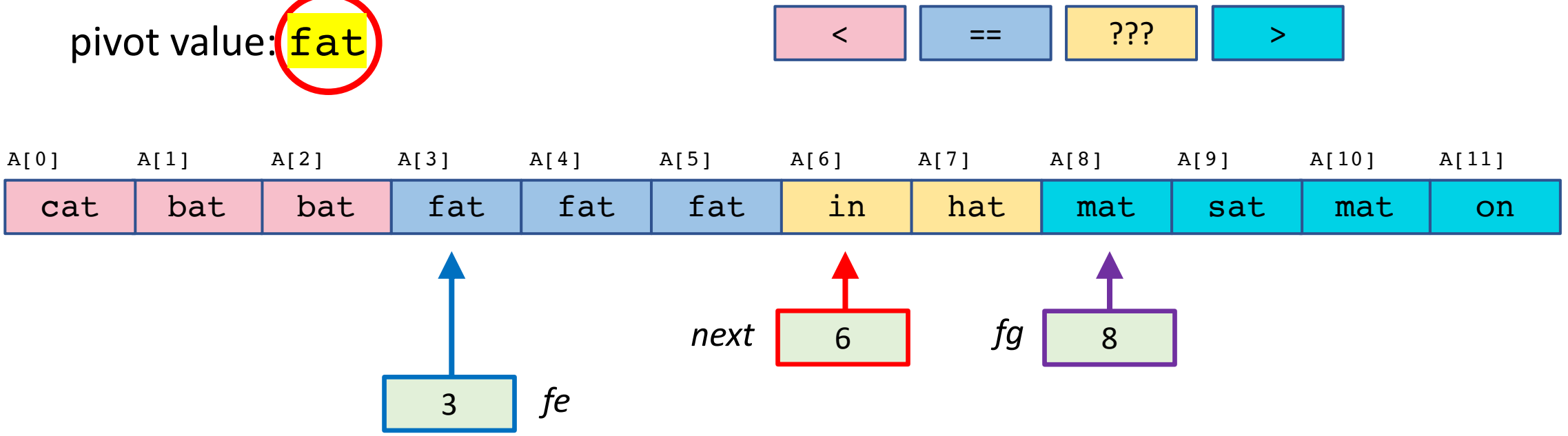
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



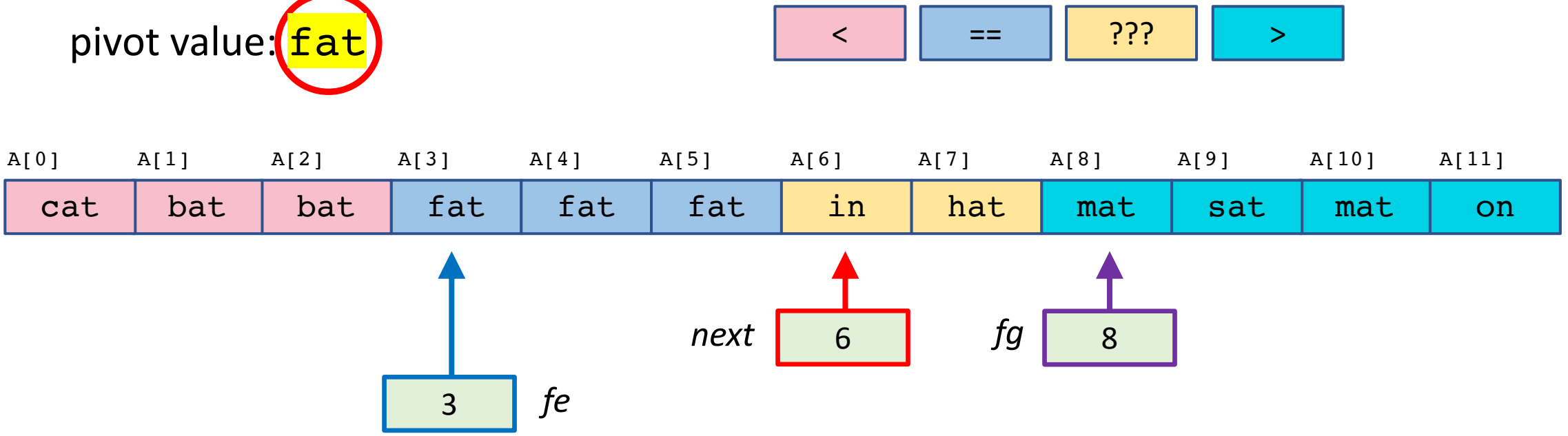
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



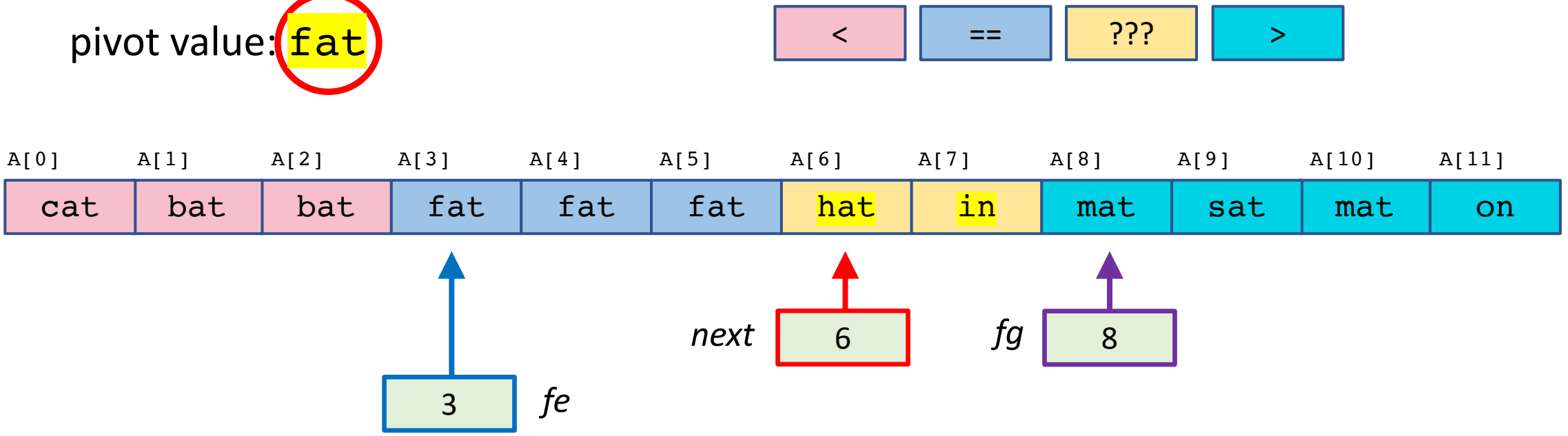
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



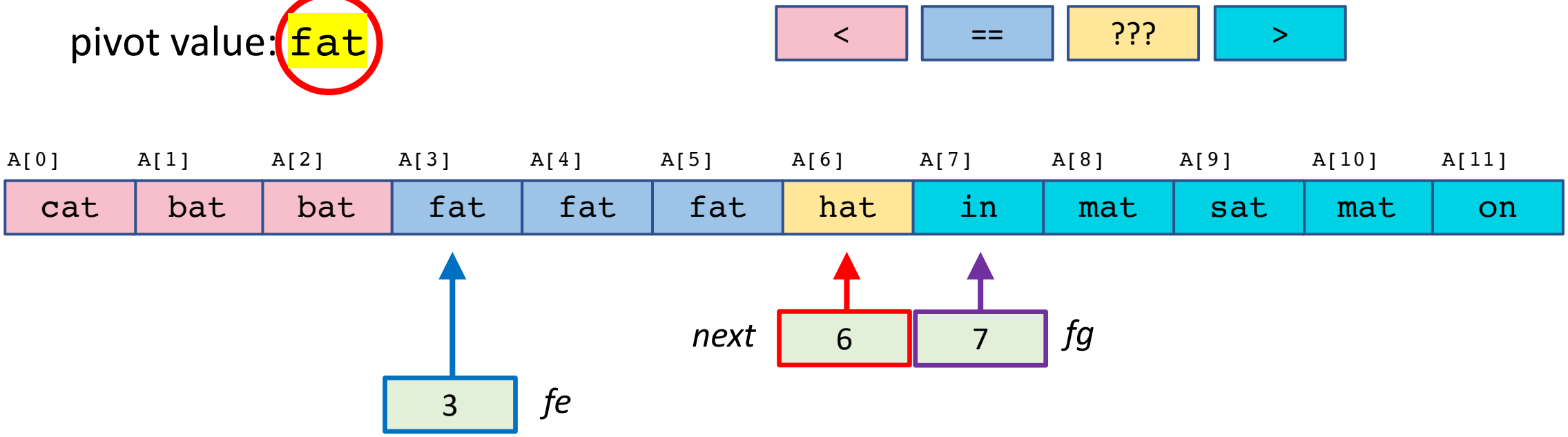
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`



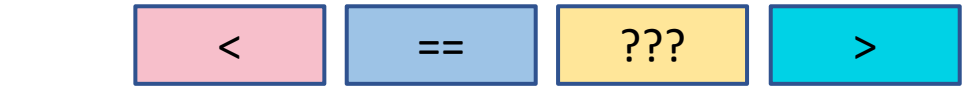

```
next, fe, fg ← 0, 0, n
while next < fg do
    if A[next] < p then
        swap A[fe] and A[next]
        fe, next ← fe + 1, next + 1
    else if A[next] > p then
        swap A[next] and A[fg − 1]
        fg ← fg − 1
    else
        next ← next + 1
```

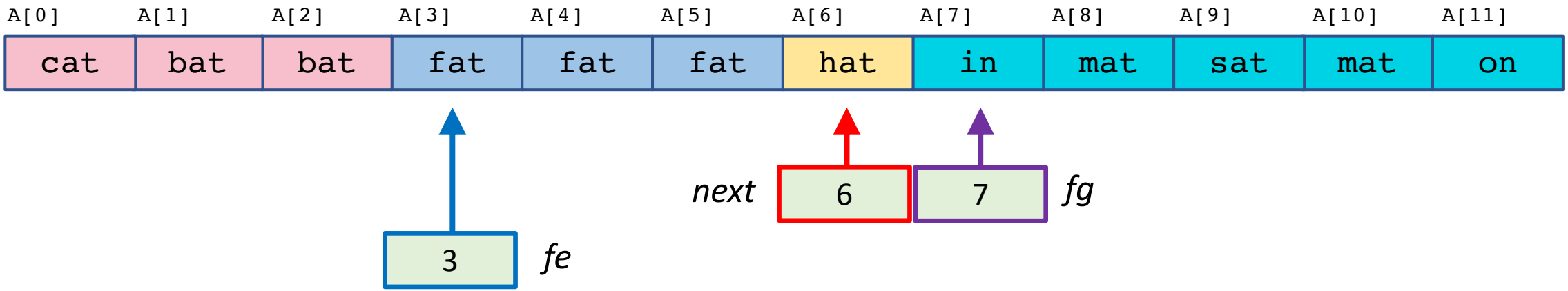
pivot value: `fat`



next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

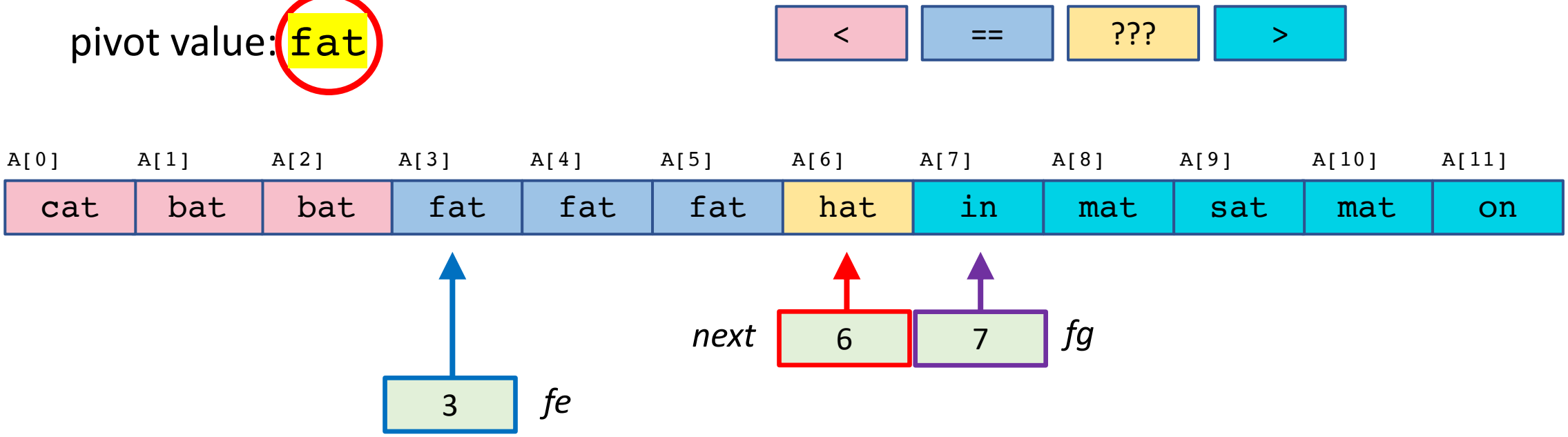
pivot value: `fat`





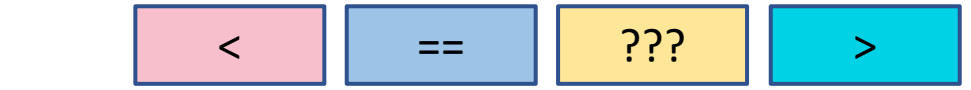
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

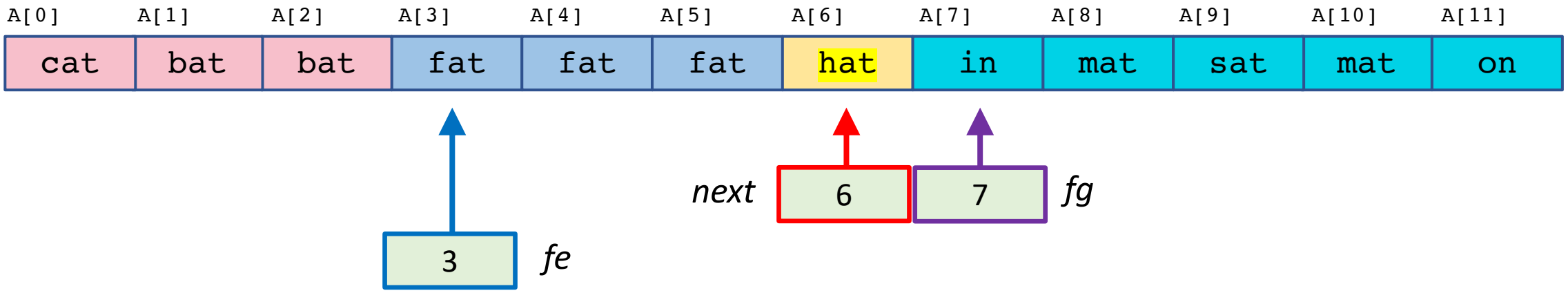
pivot value: `fat`



next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

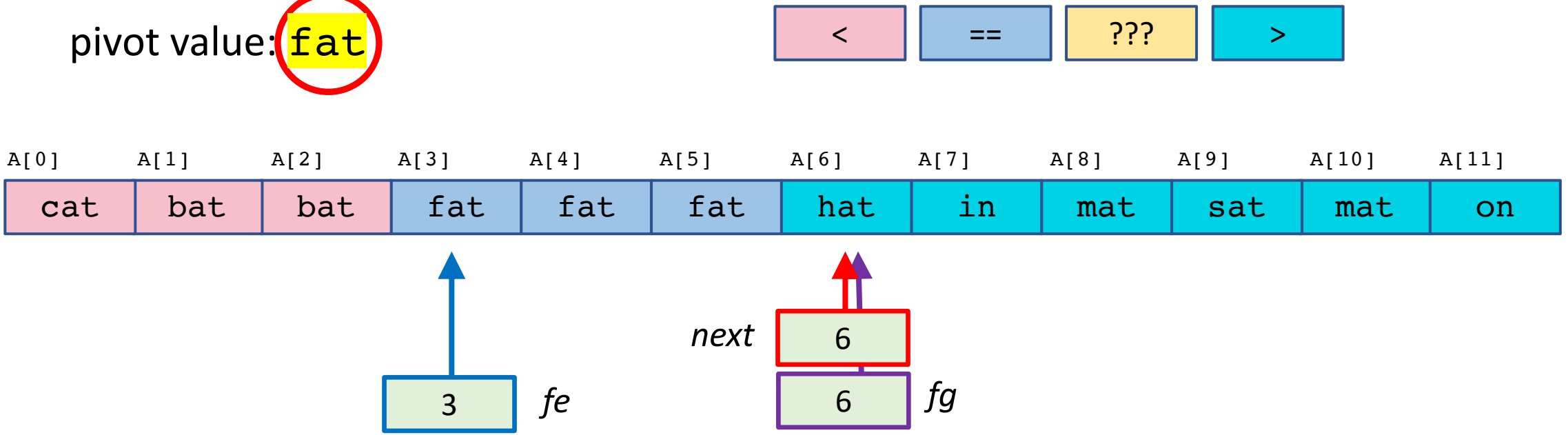
pivot value: fat





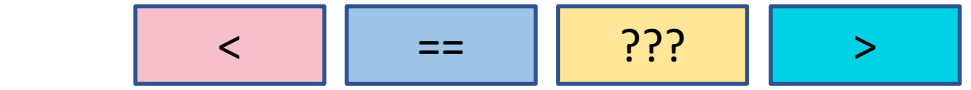
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

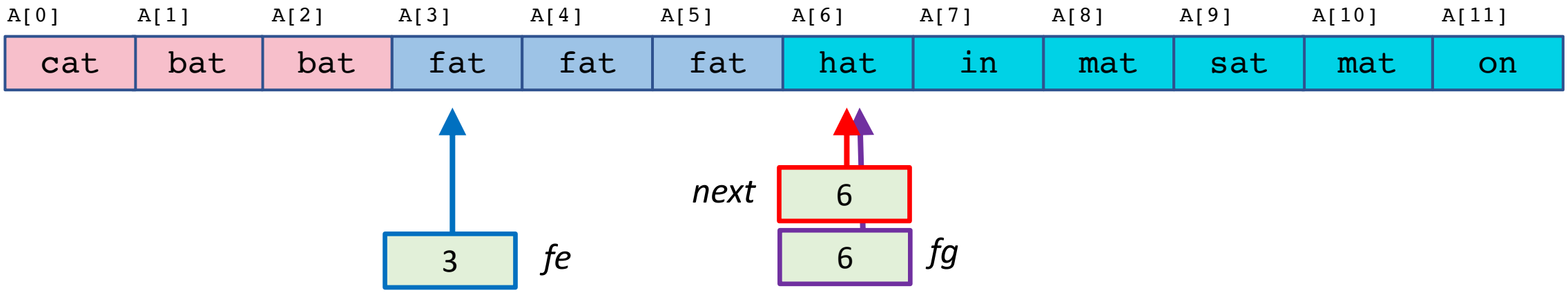
pivot value: `fat`



next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

pivot value: `fat`





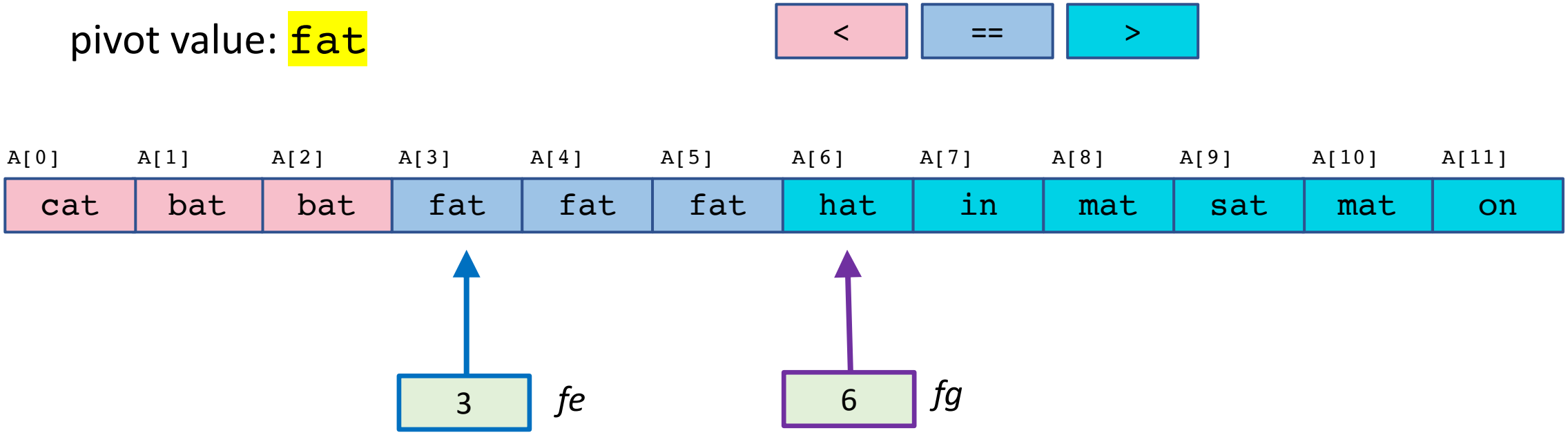
next, *fe*, *fg* ← 0, 0, *n*
while *next* < *fg* **do**
  if *A*[*next*] < *p* **then**
    swap *A*[*fe*] and *A*[*next*]
    fe, *next* ← *fe* + 1, *next* + 1
  else if *A*[*next*] > *p* **then**
    swap *A*[*next*] and *A*[*fg* − 1]
    fg ← *fg* − 1
  else
    next ← *next* + 1

now what happens?

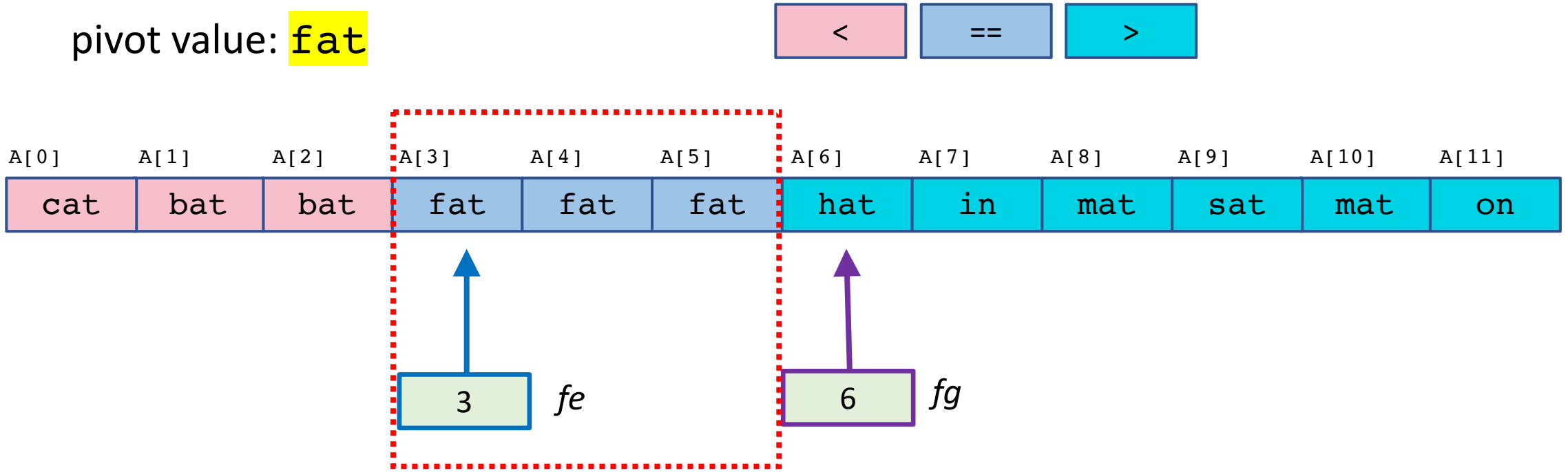
now what happens?

invisible friends!

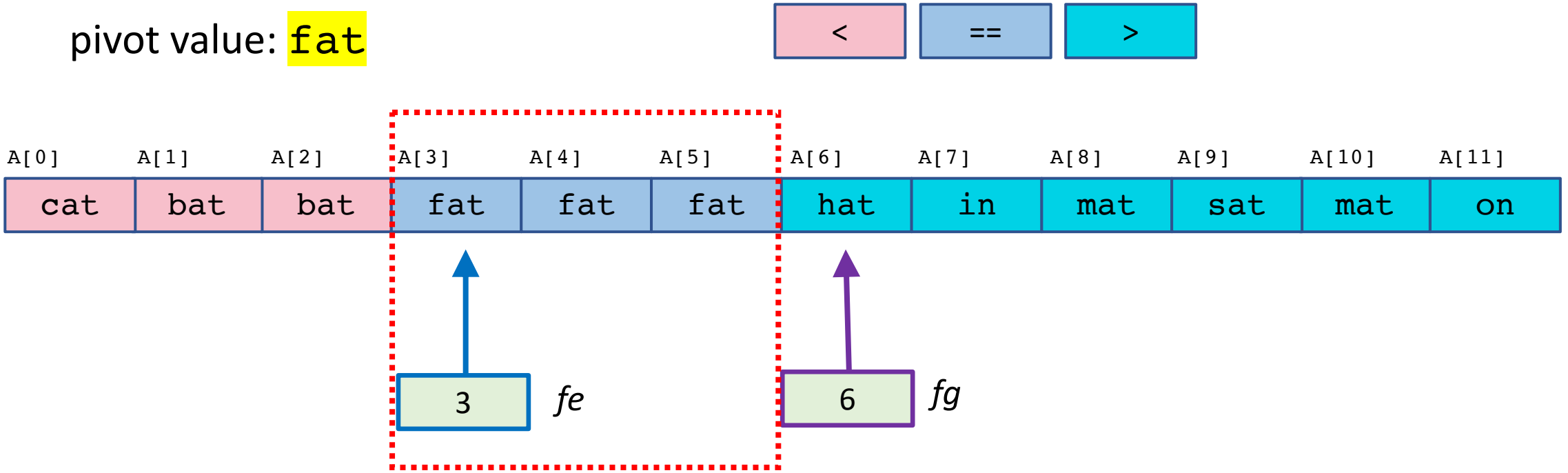
pivot value: `fat`



pivot value: `fat`

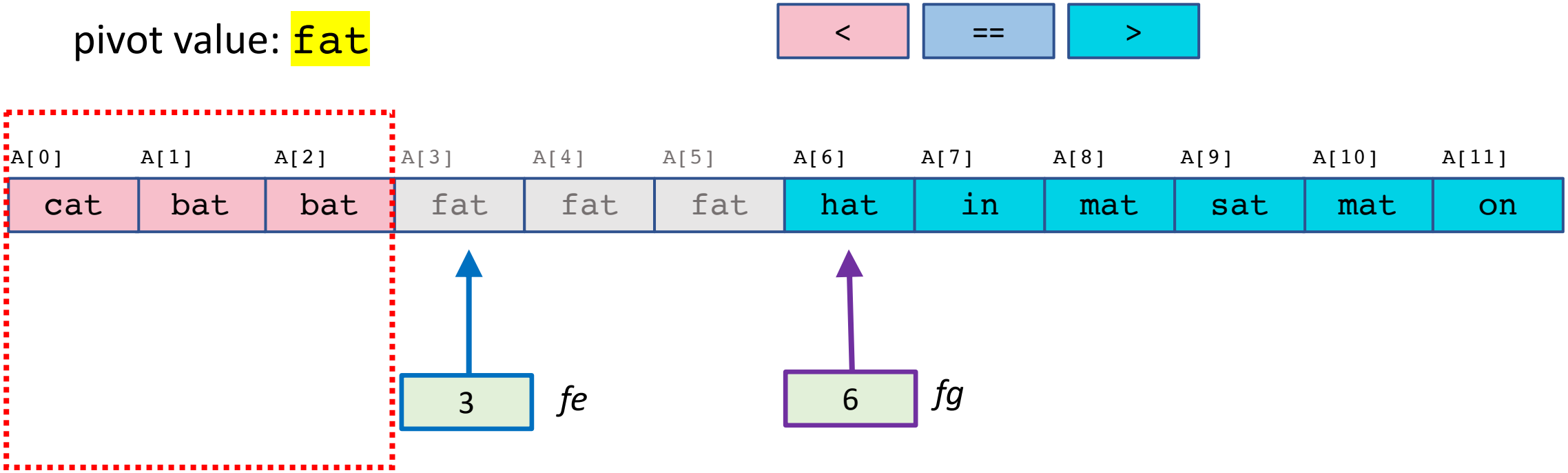


pivot value: `fat`

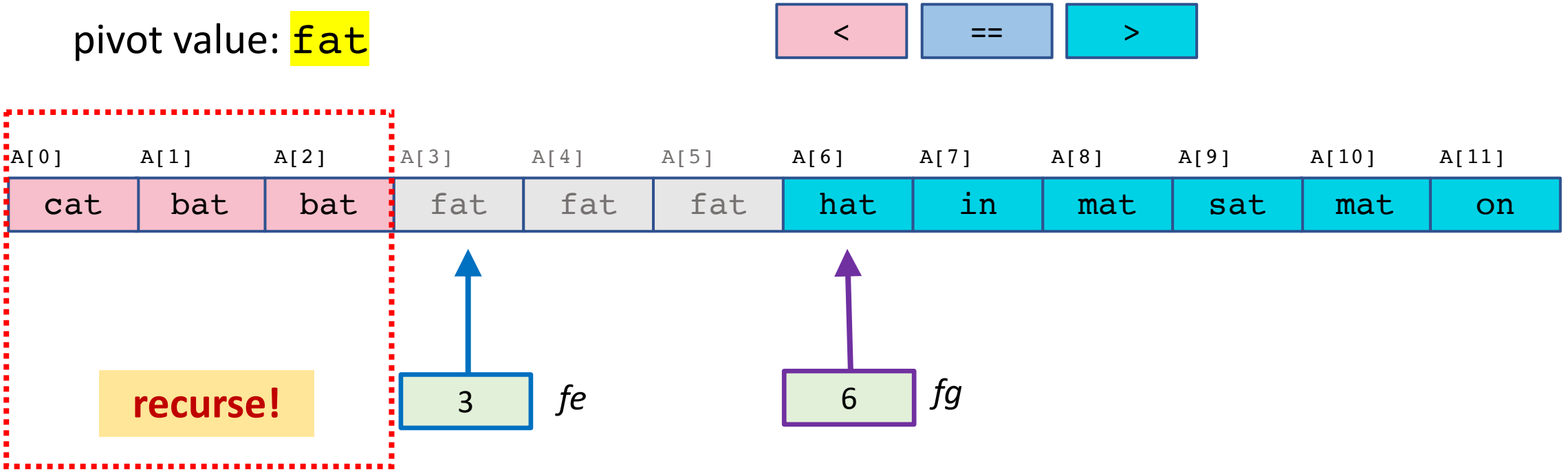


must be in correct locations, don't need to move again

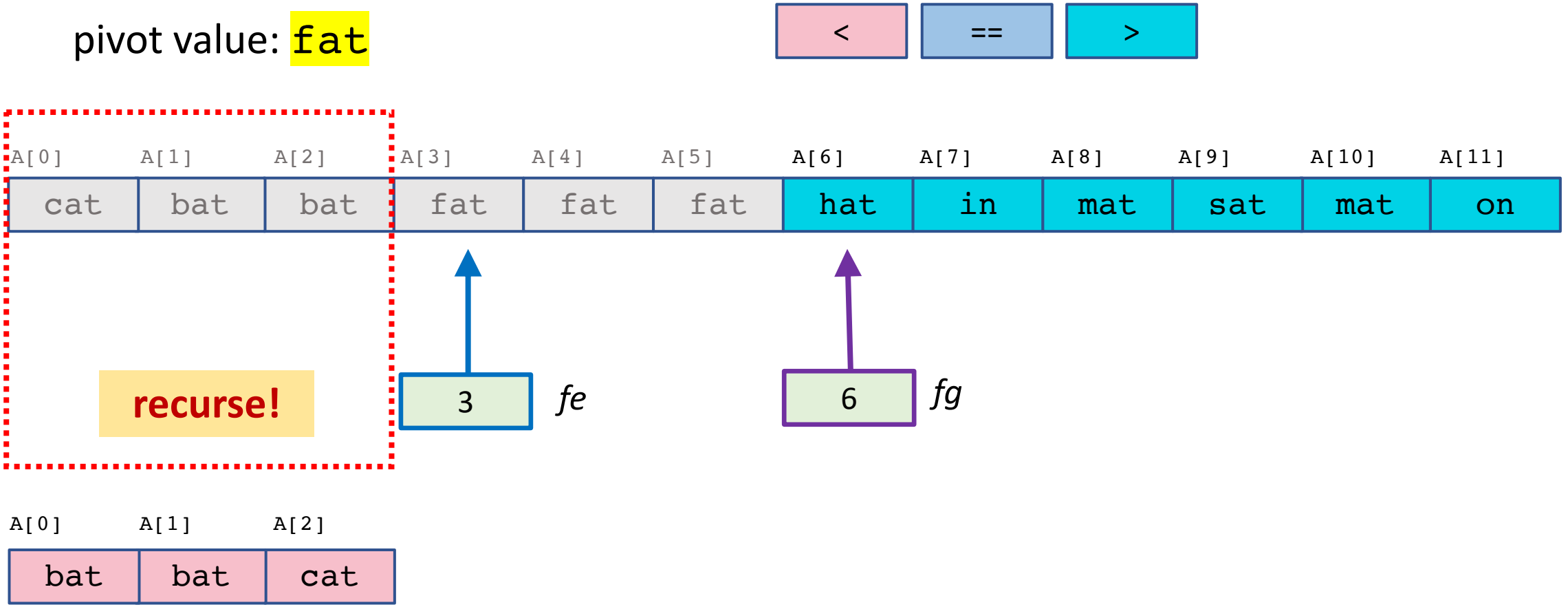
pivot value: `fat`



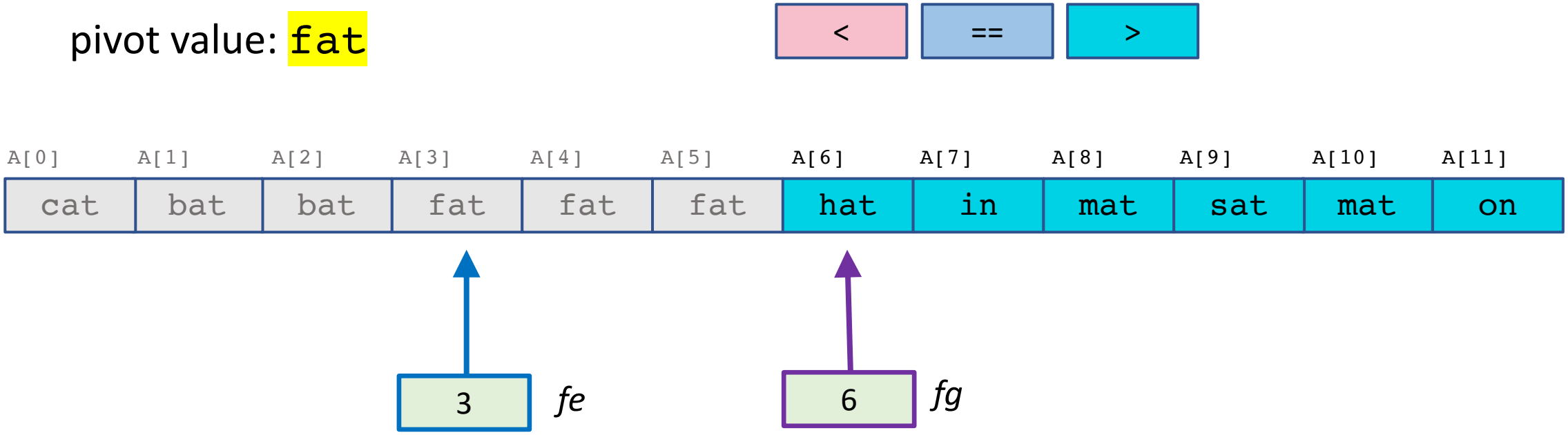
pivot value: `fat`

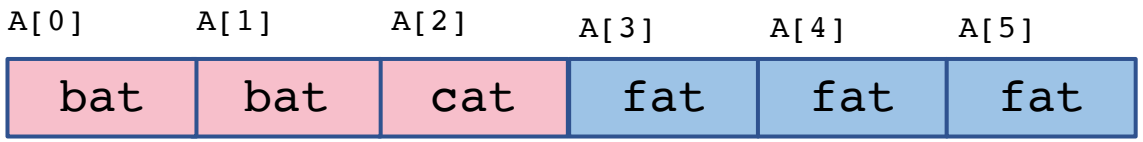


pivot value: `fat`

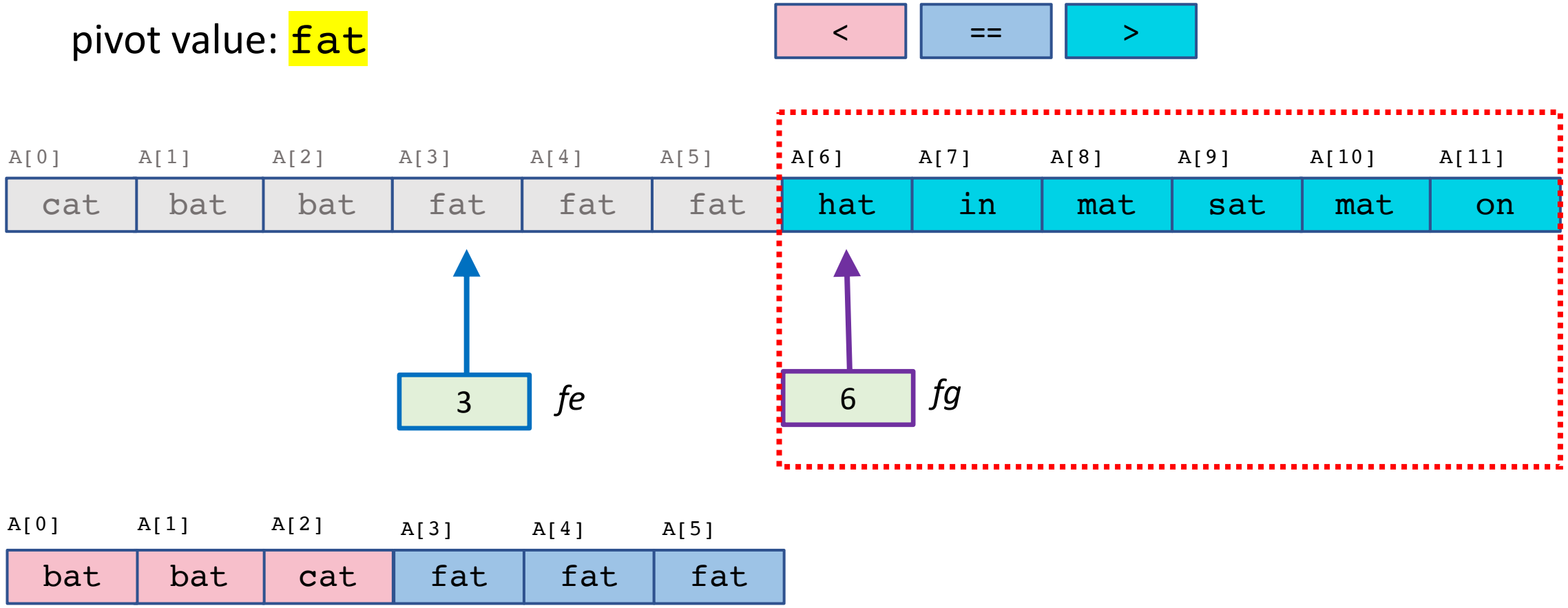


pivot value: `fat`

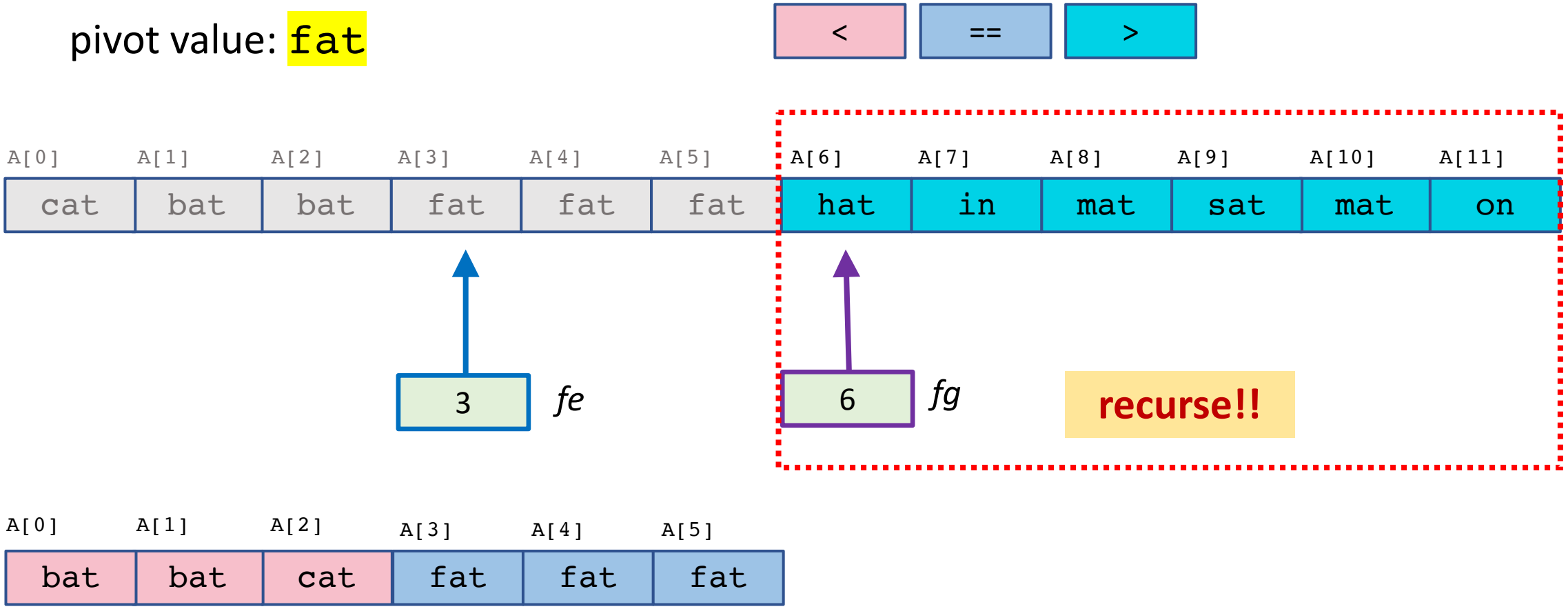




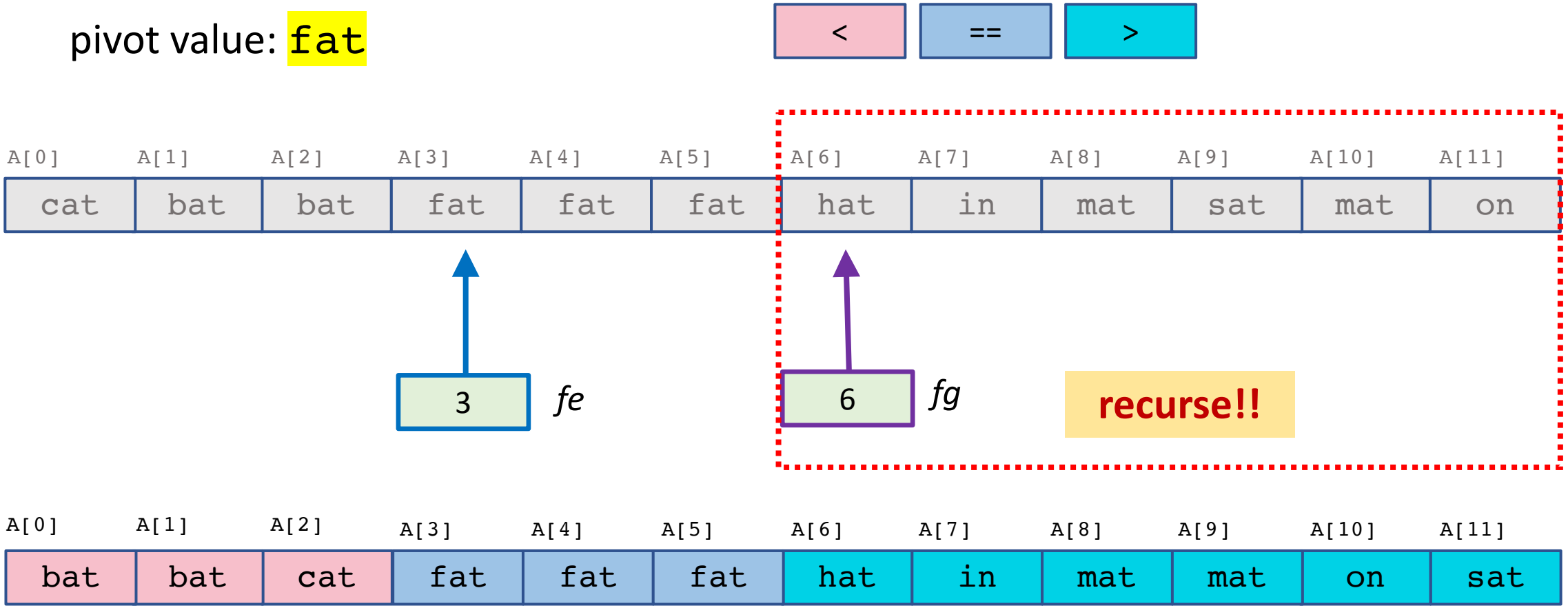
pivot value: `fat`

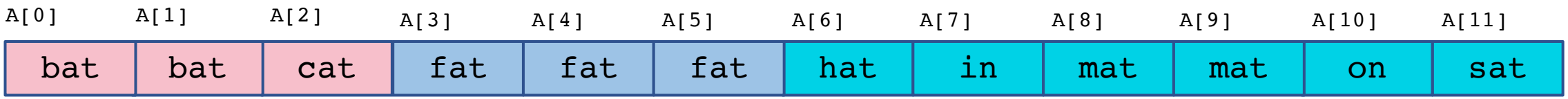


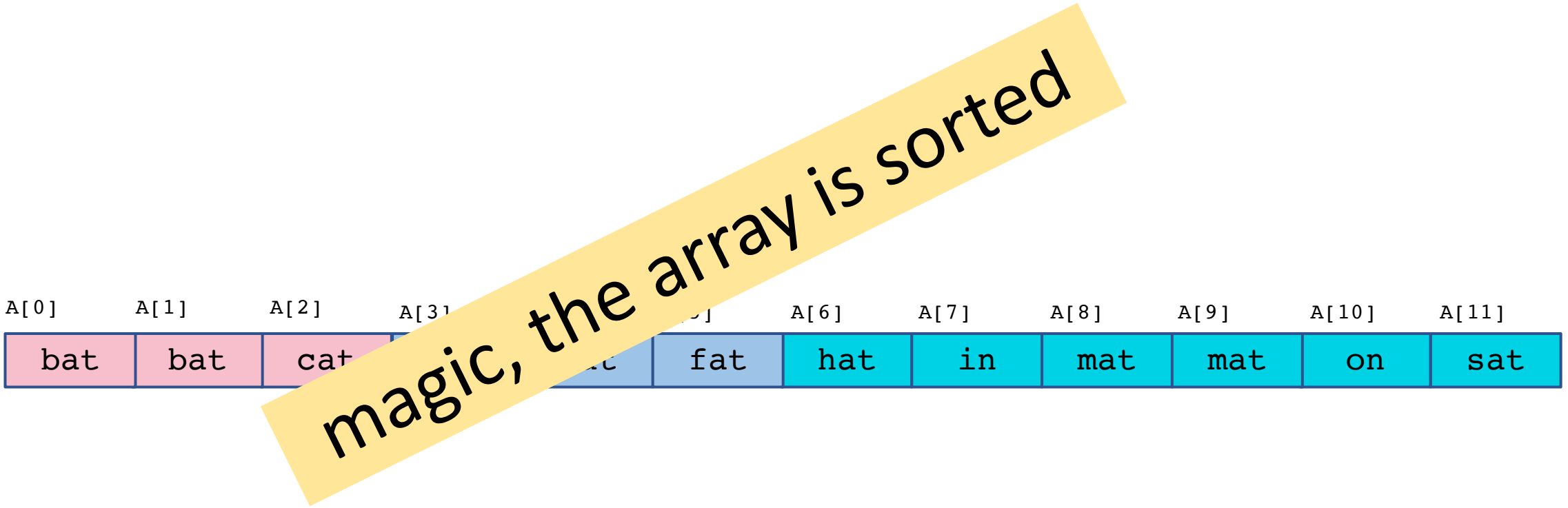
pivot value: `fat`



pivot value: `fat`



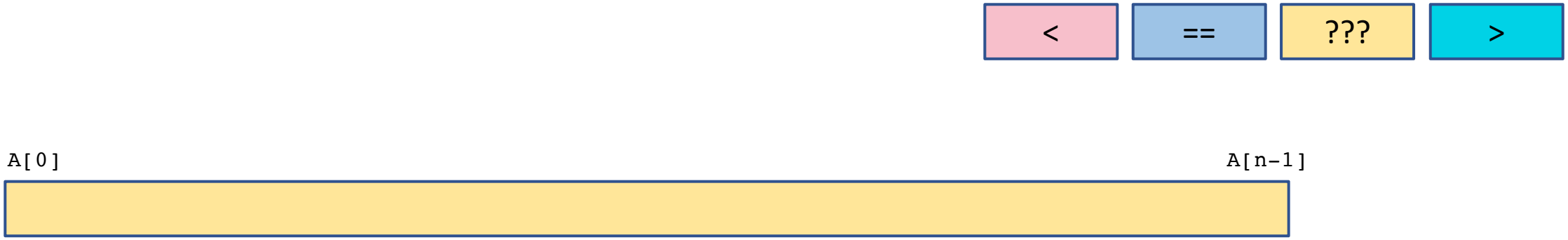

A[0]
A[1]
A[2]
A[3]
A[4]
A[5]
A[6]
A[7]
A[8]
A[9]
A[10]
A[11]
bat
bat
cat
fat
fat
fat
hat
in
mat
mat
on
sat


magic, the array is sorted
A[0]
A[1]
A[2]
A[3]
A[6]
A[7]
A[8]
A[9]
A[10]
A[11]
bat
bat
cat
fat
hat
in
mat
mat
on
sat

magic, the array is sorted

thanks, invisible friends!

now for the big picture



quicksort(*A*, *n*):

if *n* > 1 **then**

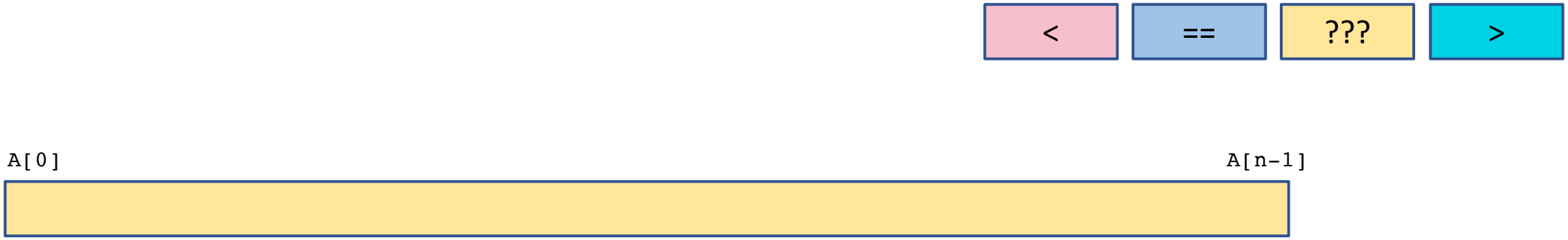
p ← any element in *A*[0 .. *n* − 1]

(*fe*, *fg*) ← partition(*A*, *n*, *p*)

quicksort(*A*, *fe*)

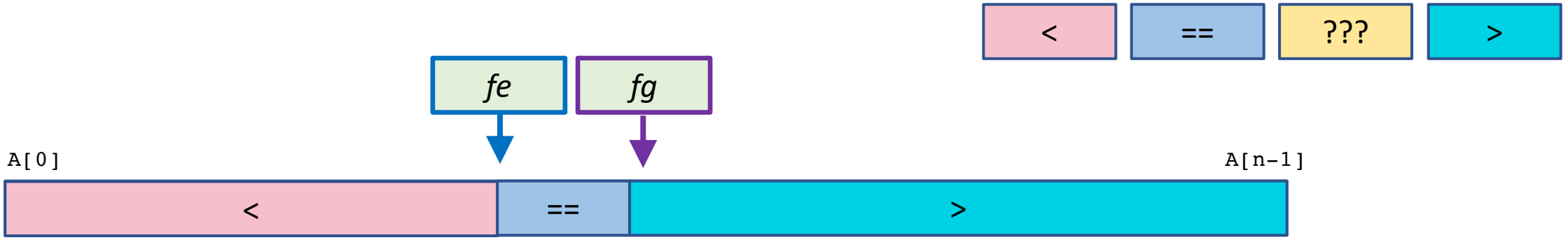
quicksort(*A* + *fg*, *n* − *fg*)

return

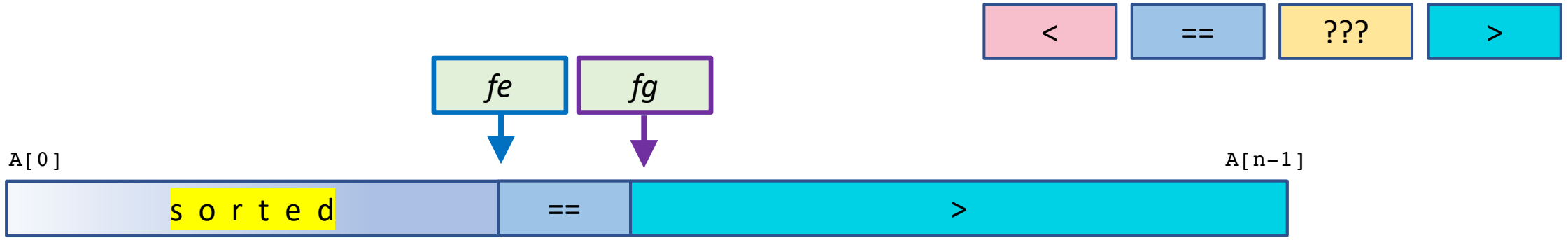


quicksort(*A*, *n*):
  if *n* > 1 **then**
    p ← any element in *A*[0 .. *n* − 1]
    (*fe*, *fg*) ← partition(*A*, *n*, *p*)
    quicksort(*A*, *fe*)
    quicksort(*A* + *fg*, *n* − *fg*)
  return

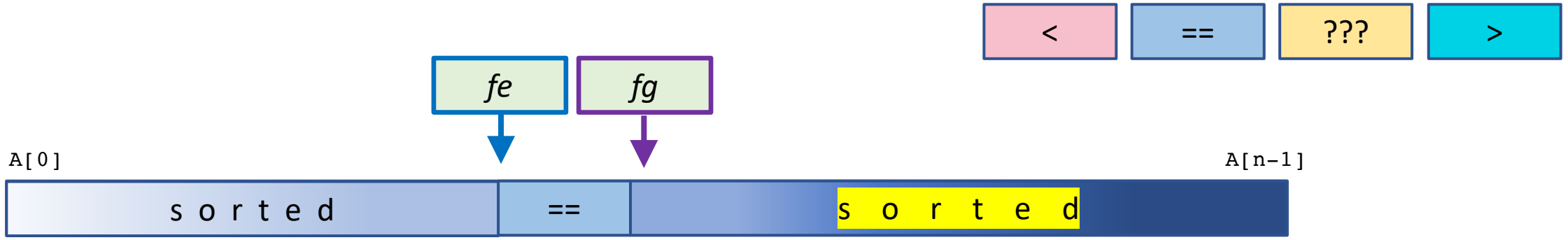
quicksort(*A*, *n*):
  if *n* > 1 **then**
    p ← any element in *A*[0 .. *n* − 1]
    (*fe*, *fg*) ← partition(*A*, *n*, *p*)
    quicksort(*A*, *fe*)
    quicksort(*A* + *fg*, *n* − *fg*)
  return




```
quicksort(A, n):
    if n > 1 then
        p ← any element in A[0 .. n − 1]
        (fe, fg) ← partition(A, n, p)
        quicksort(A, fe)
        quicksort(A + fg, n − fg)
    return
```



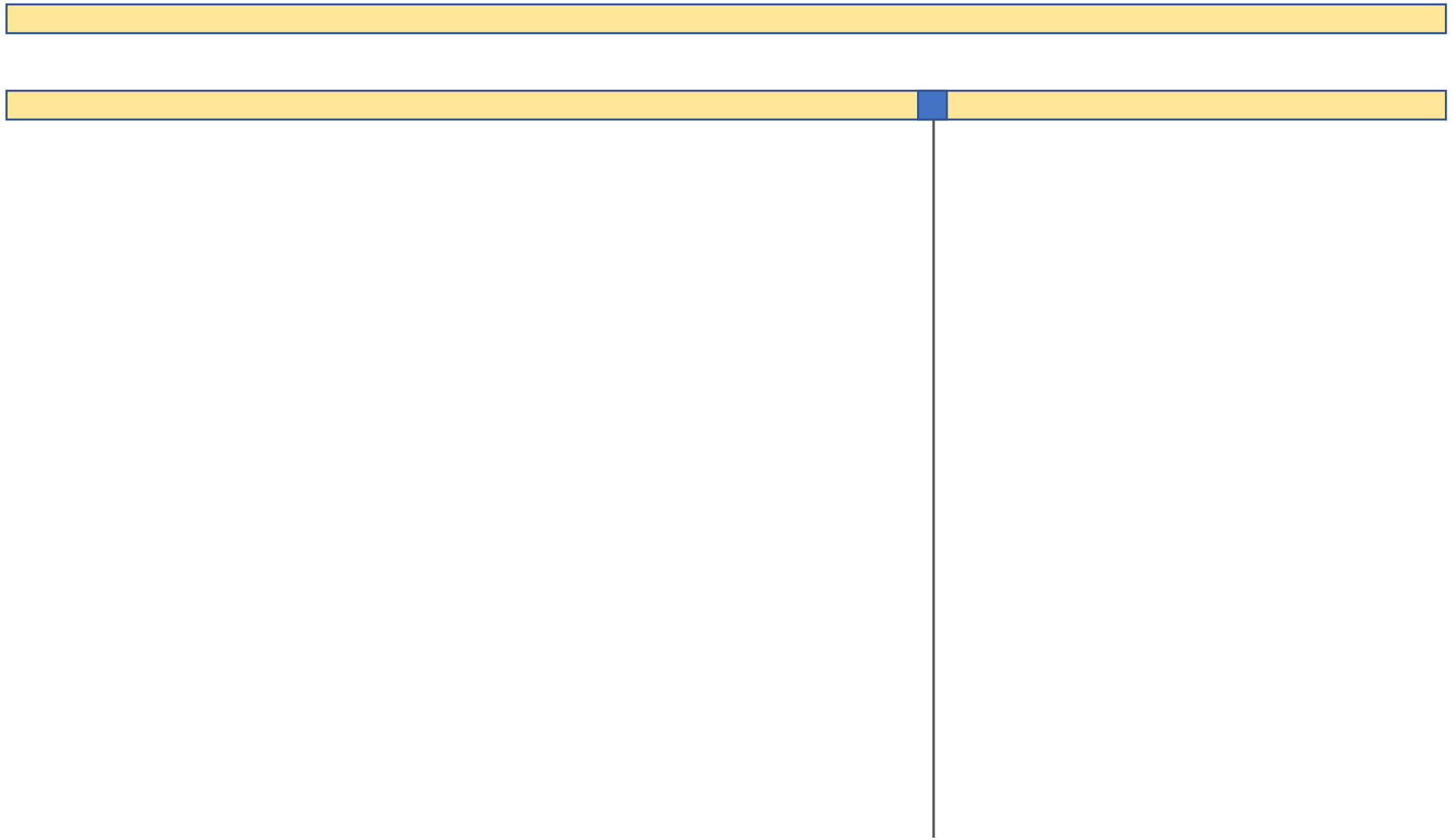
quicksort(*A*, *n*):
  if *n* > 1 **then**
    p ← any element in *A*[0 .. *n* − 1]
    (*fe*, *fg*) ← partition(*A*, *n*, *p*)
    quicksort(*A*, *fe*)
    quicksort(*A* + *fg*, *n* − *fg*)
  return

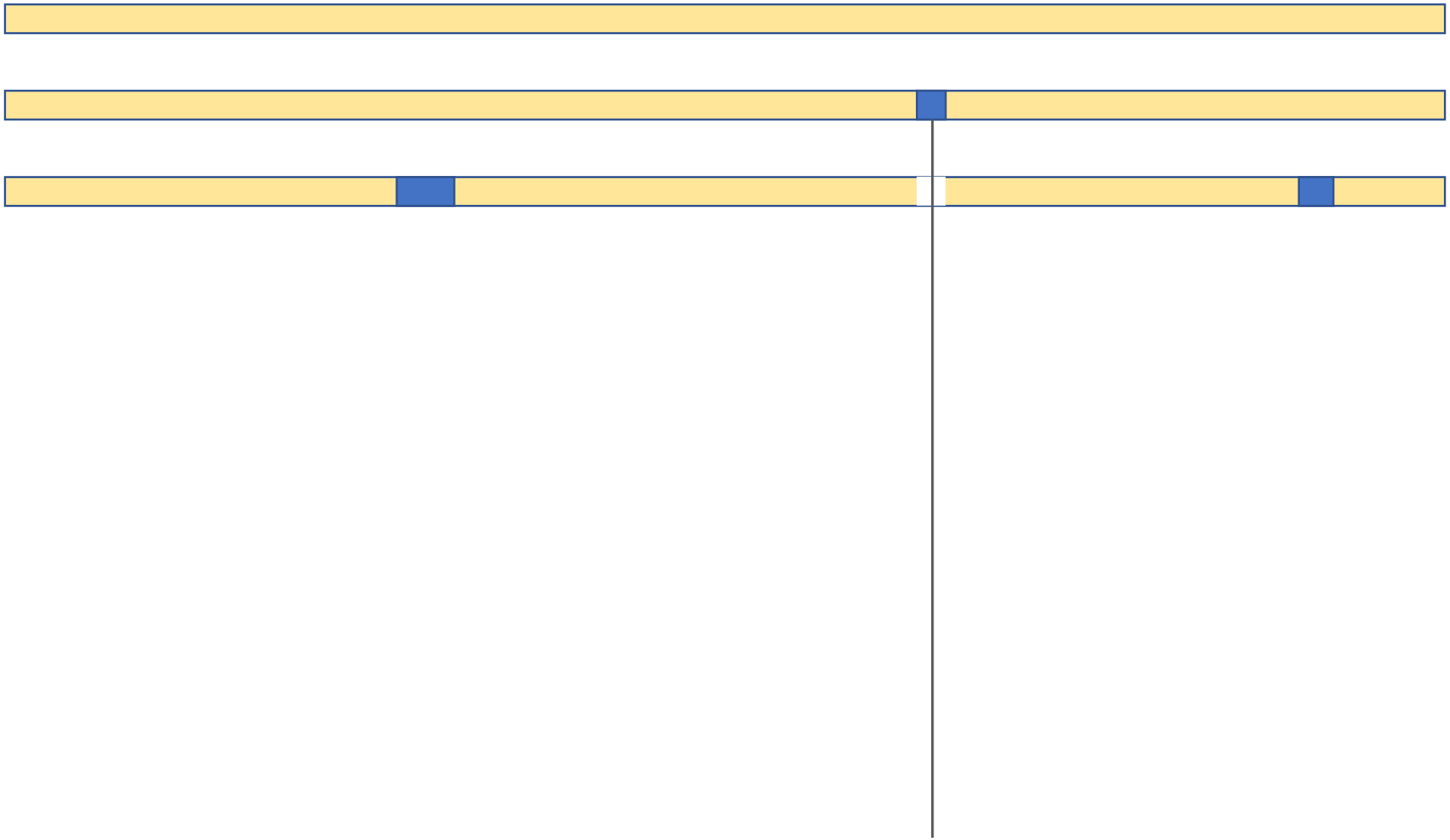


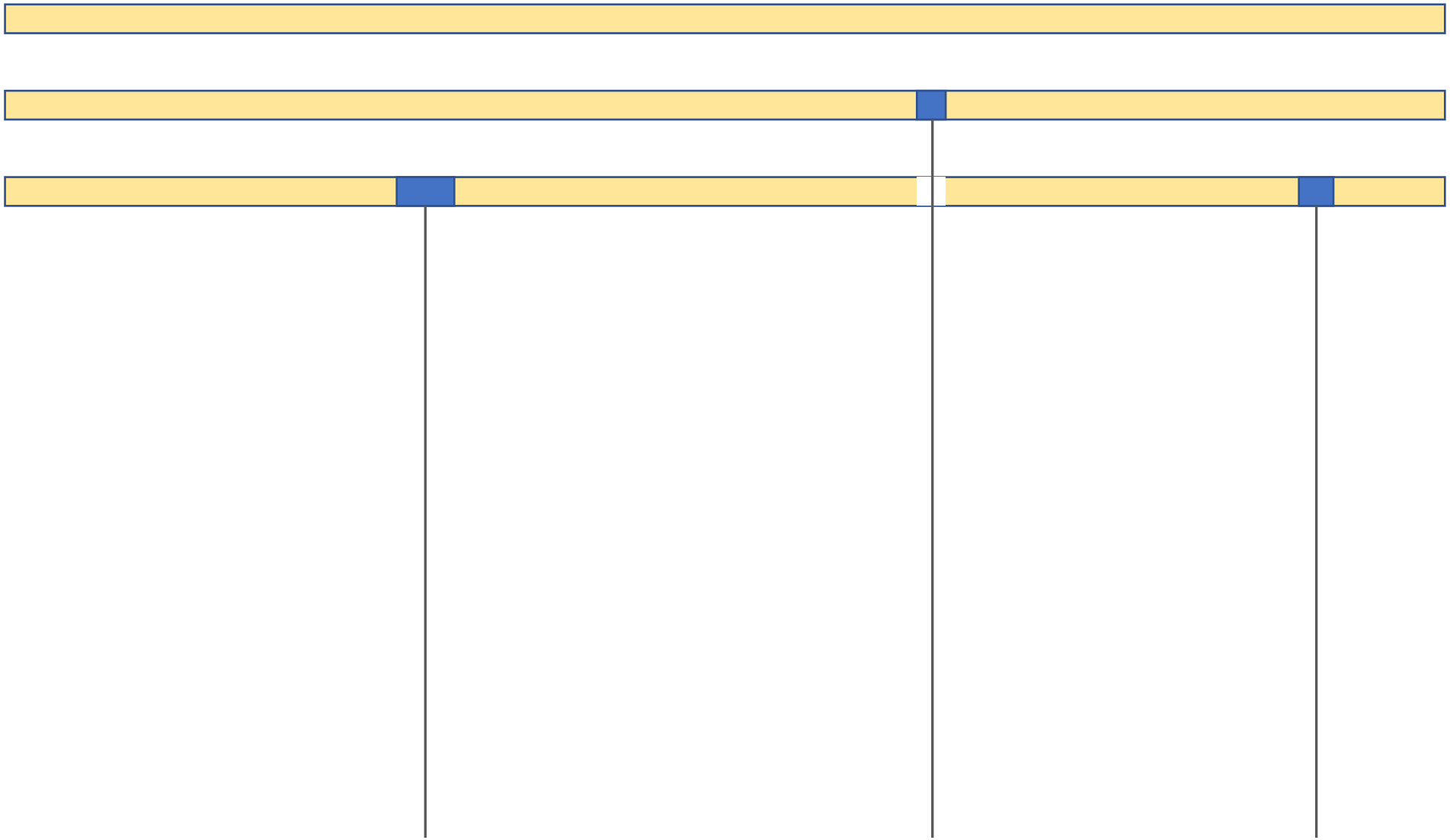
quicksort(*A*, *n*):
  if *n* > 1 **then**
    p ← any element in *A*[0 .. *n* − 1]
    (*fe*, *fg*) ← partition(*A*, *n*, *p*)
    quicksort(*A*, *fe*)
    quicksort(*A* + *fg*, *n* − *fg*)
  return

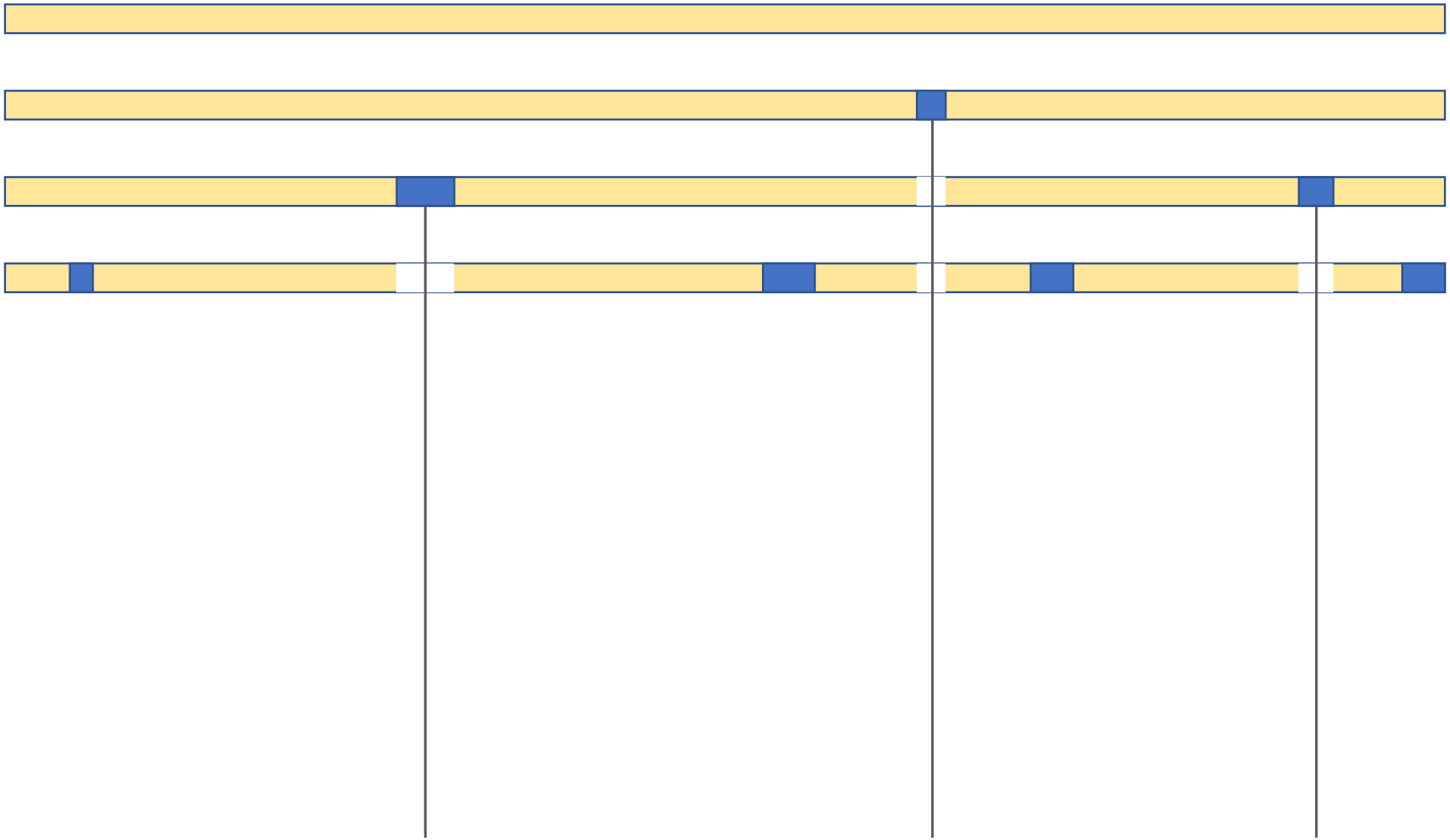
quicksort(*A*, *n*):
  if *n* > 1 **then**
    p ← any element in *A*[0 .. *n* − 1]
    (*fe*, *fg*) ← partition(*A*, *n*, *p*)
    quicksort(*A*, *fe*)
    quicksort(*A* + *fg*, *n* − *fg*)
  return

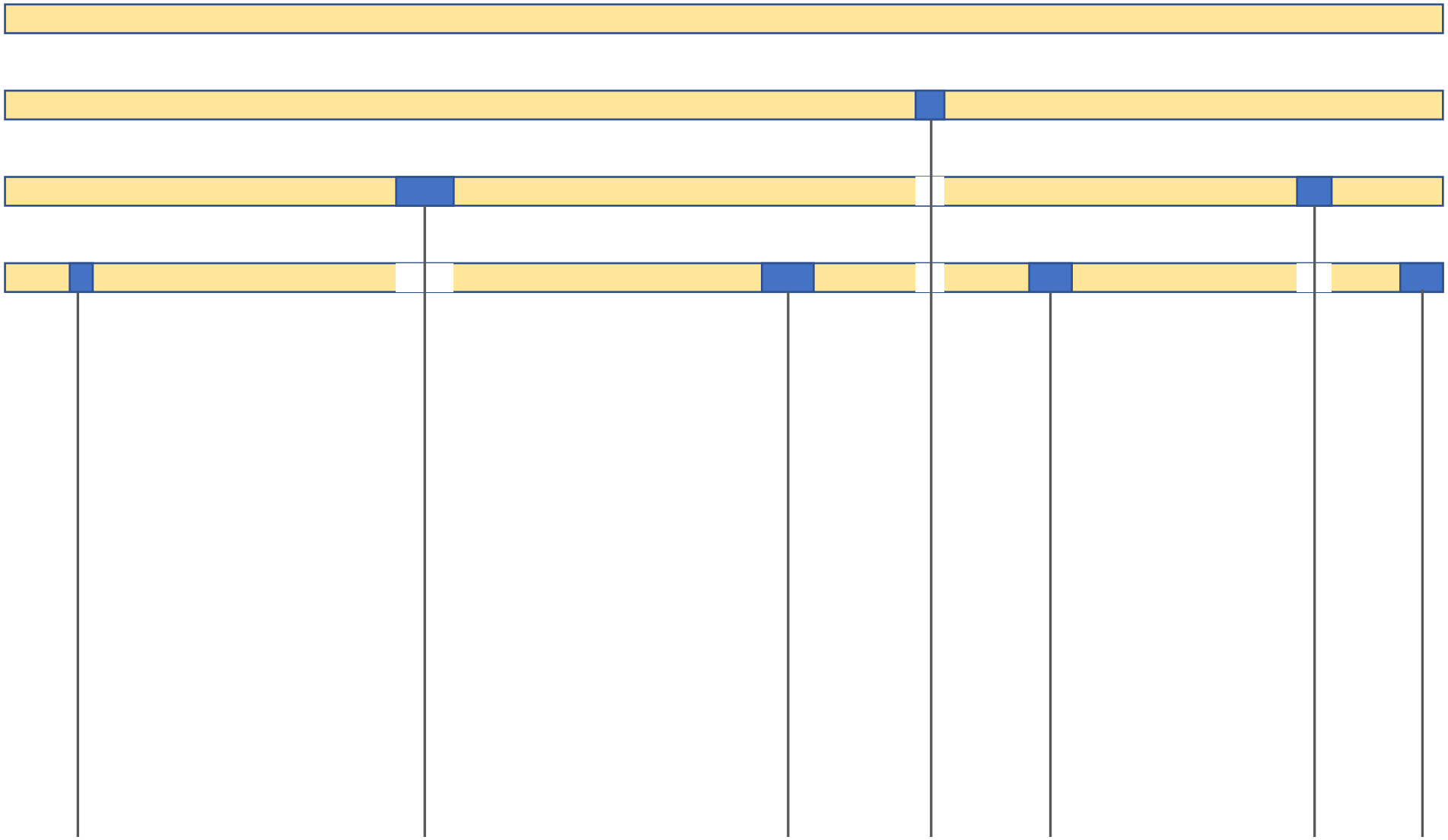
quicksort
execution time??

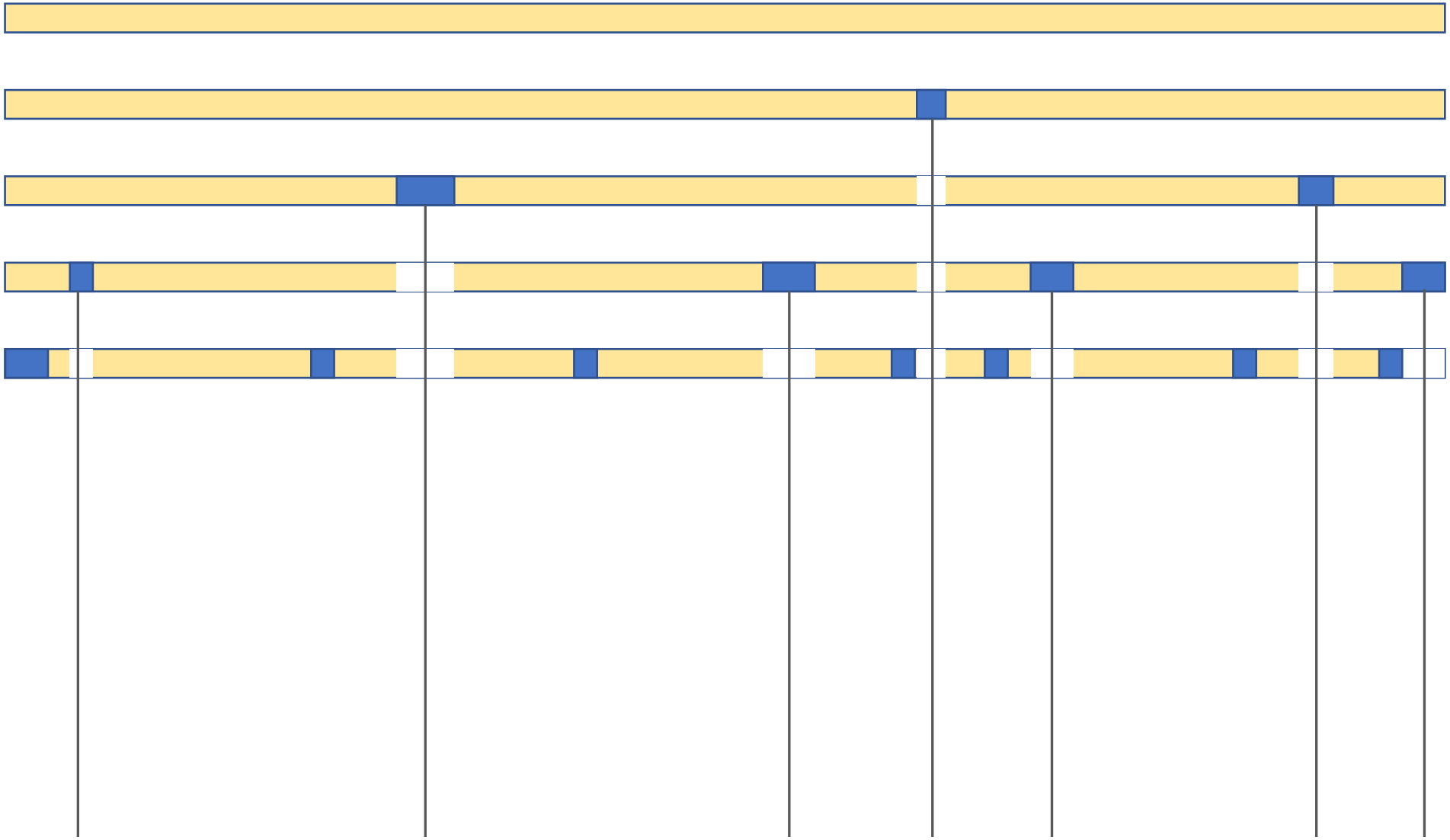


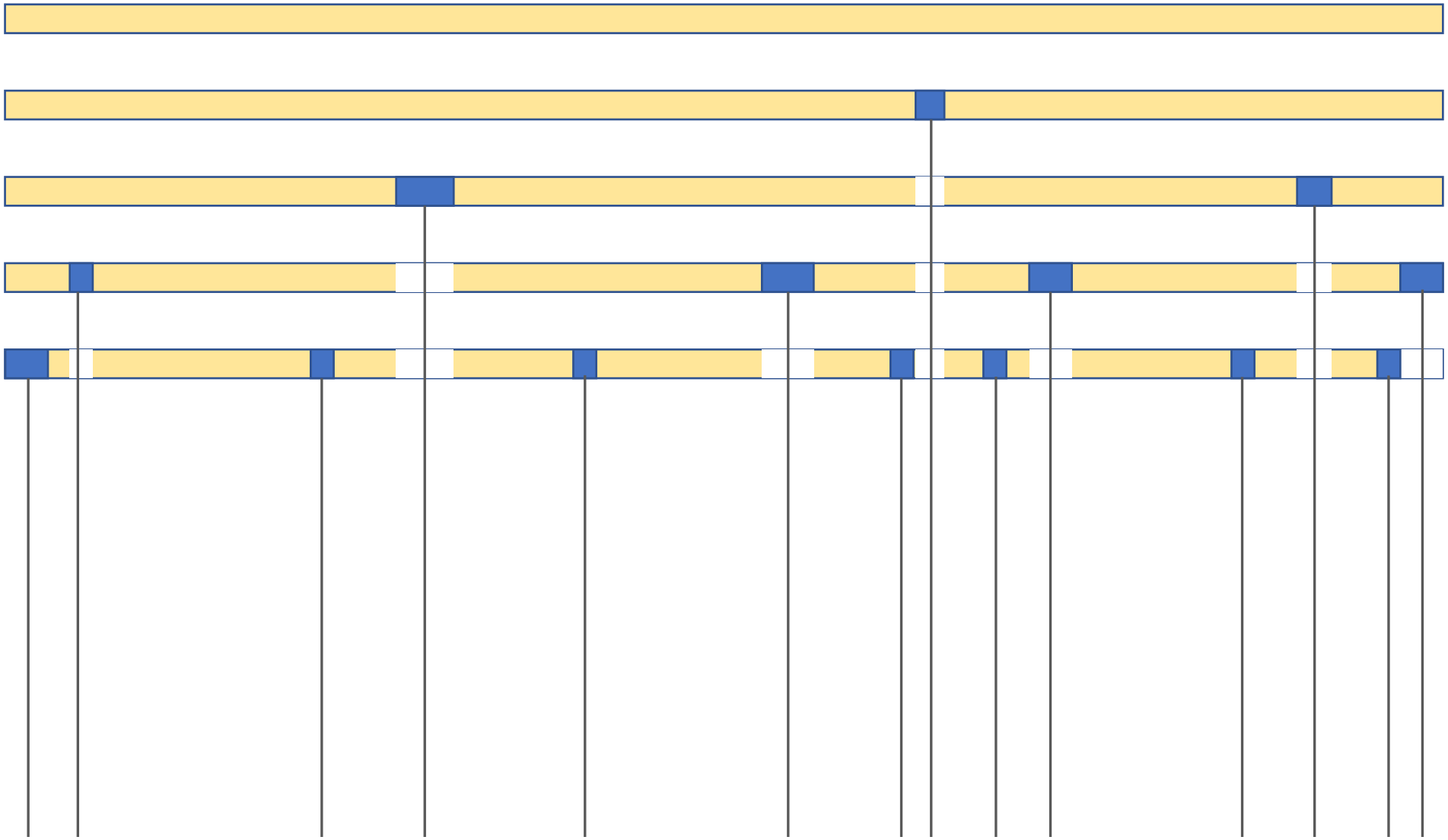


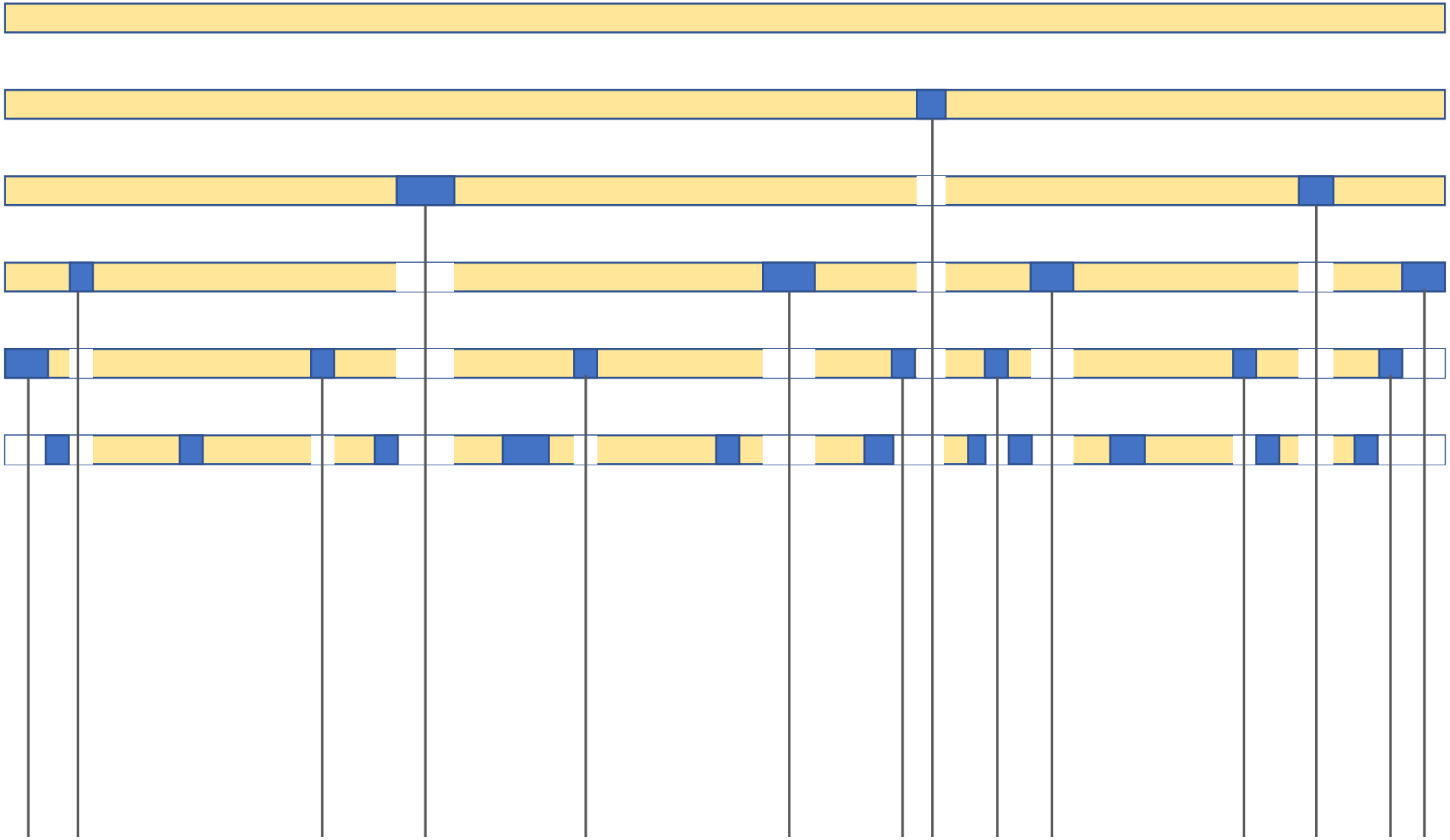


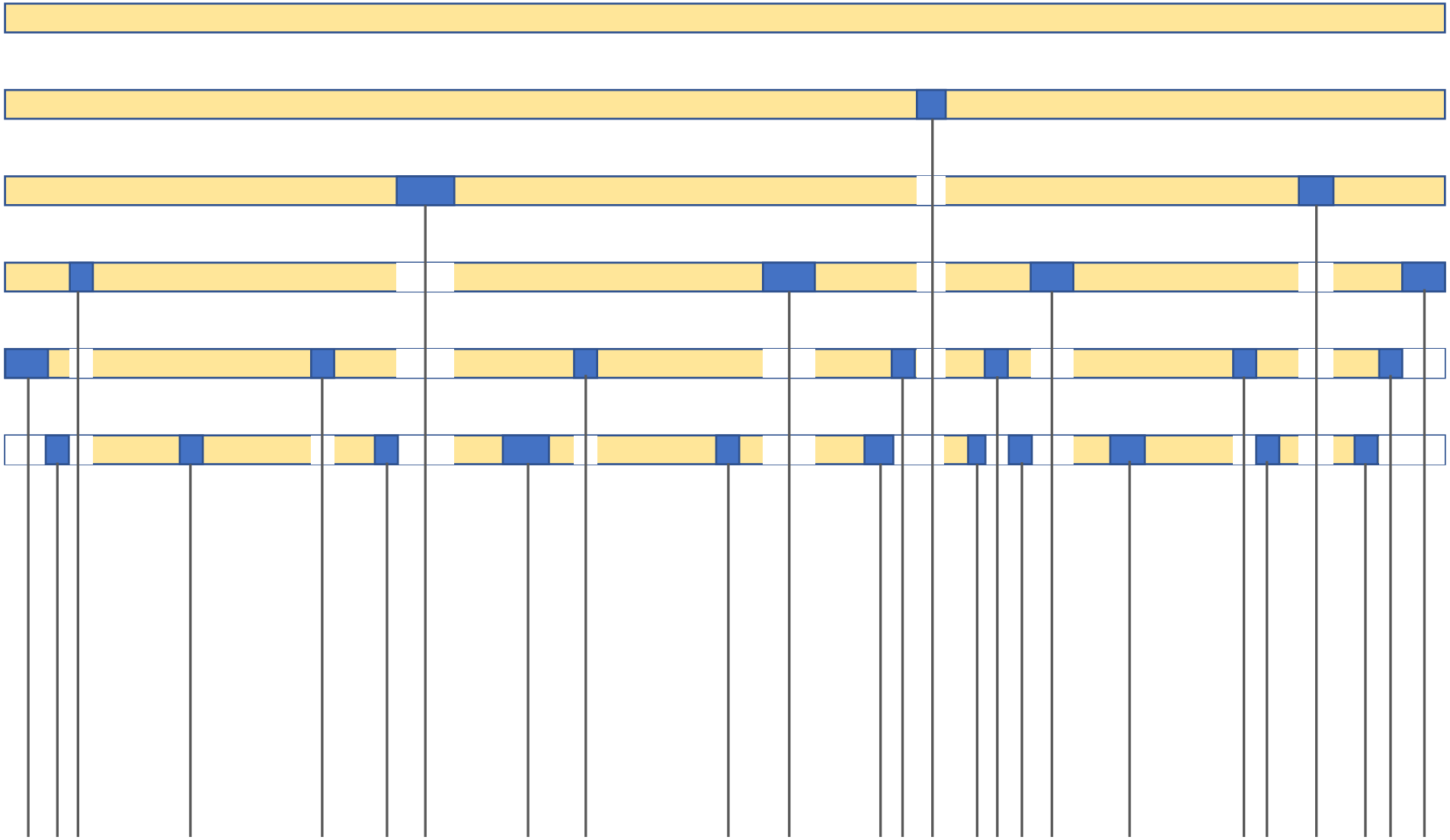


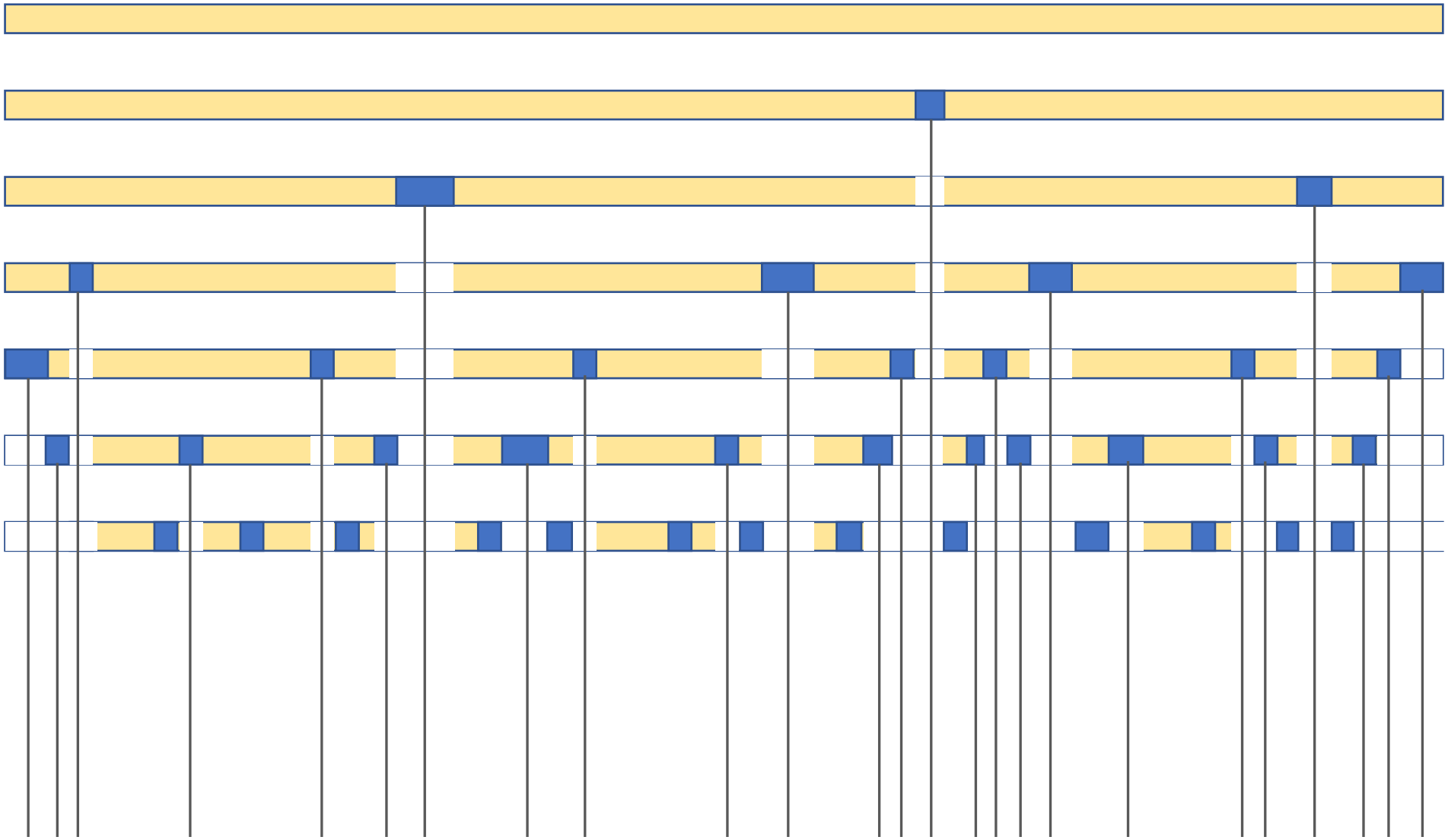


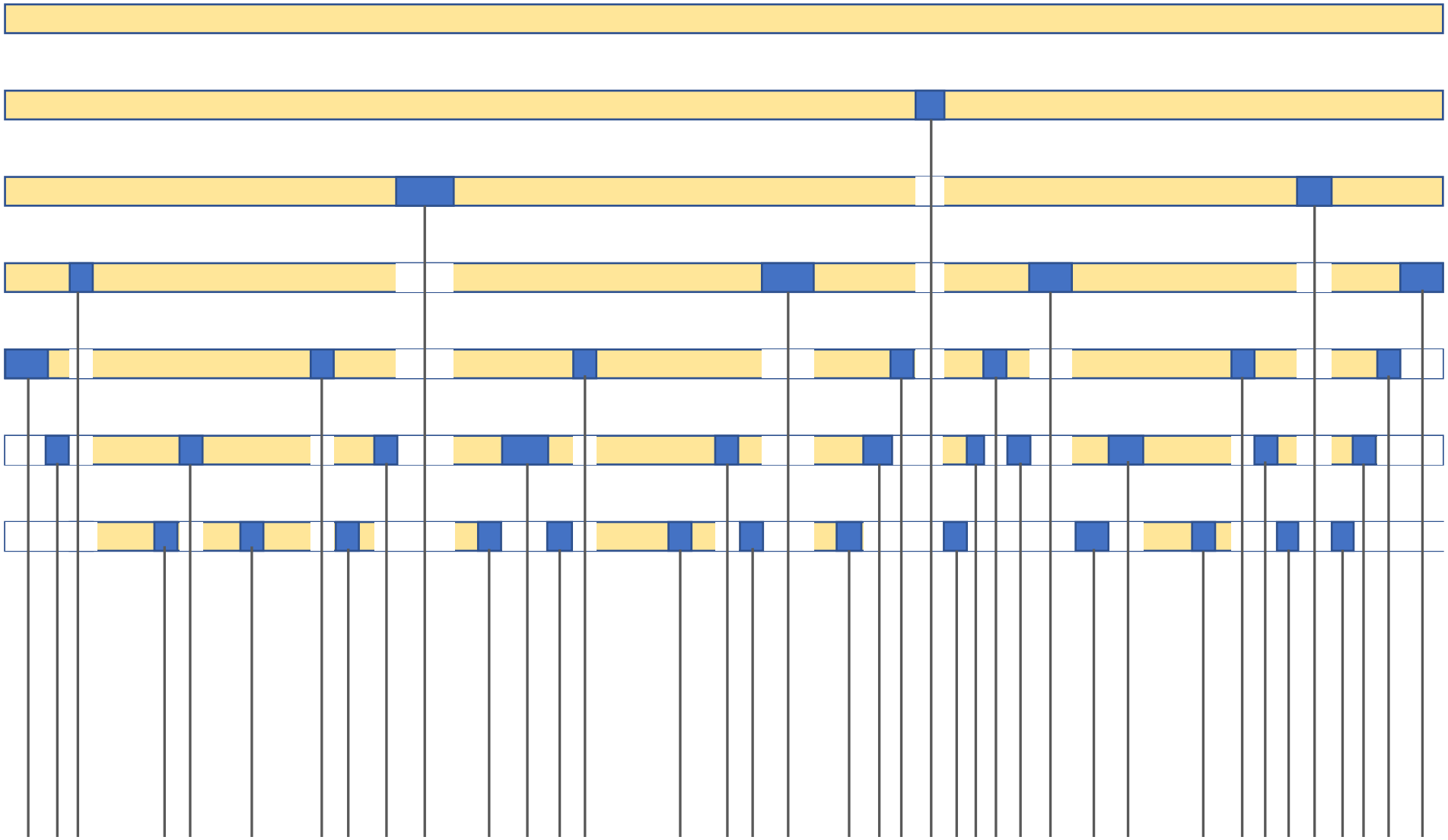


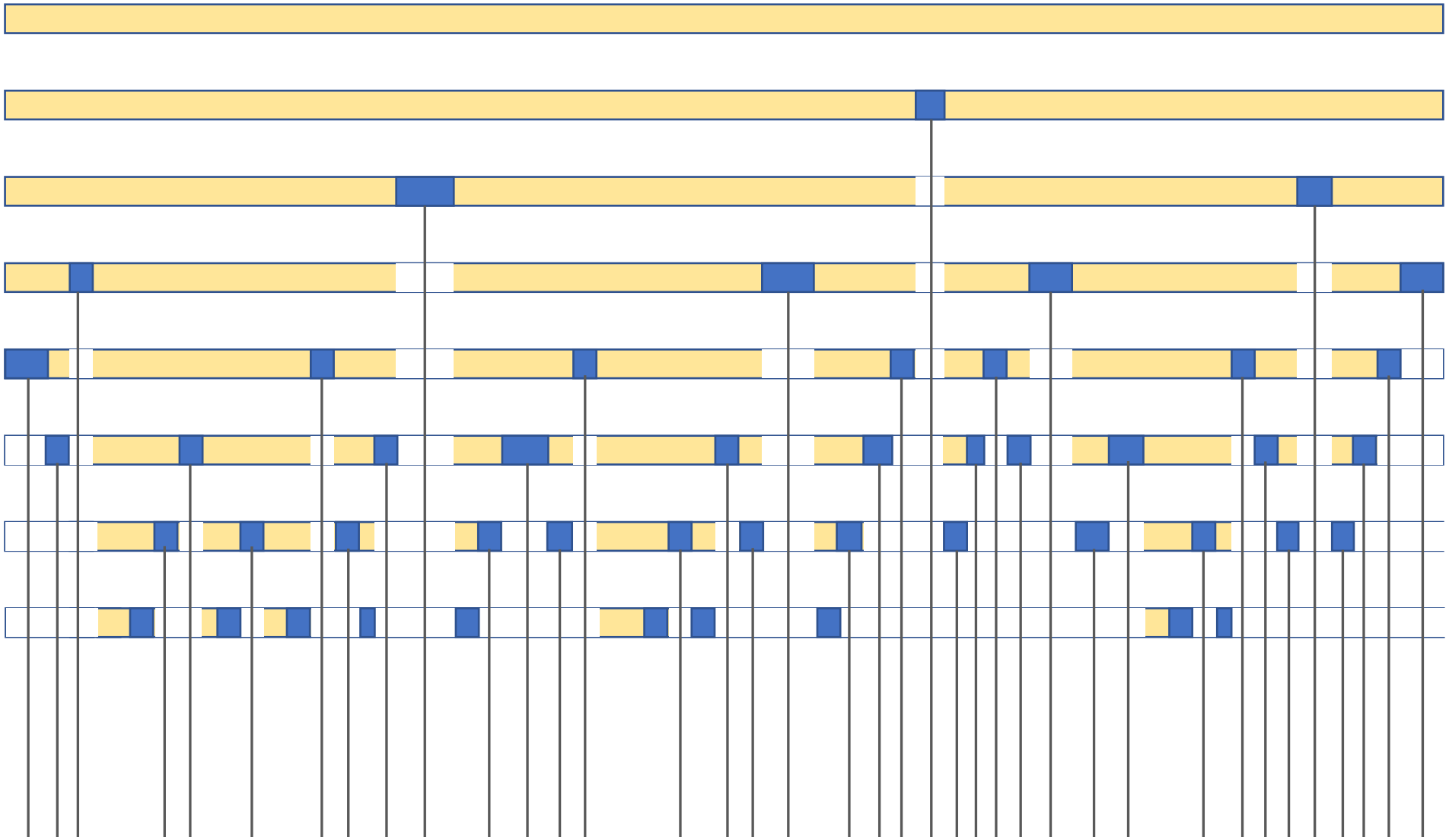


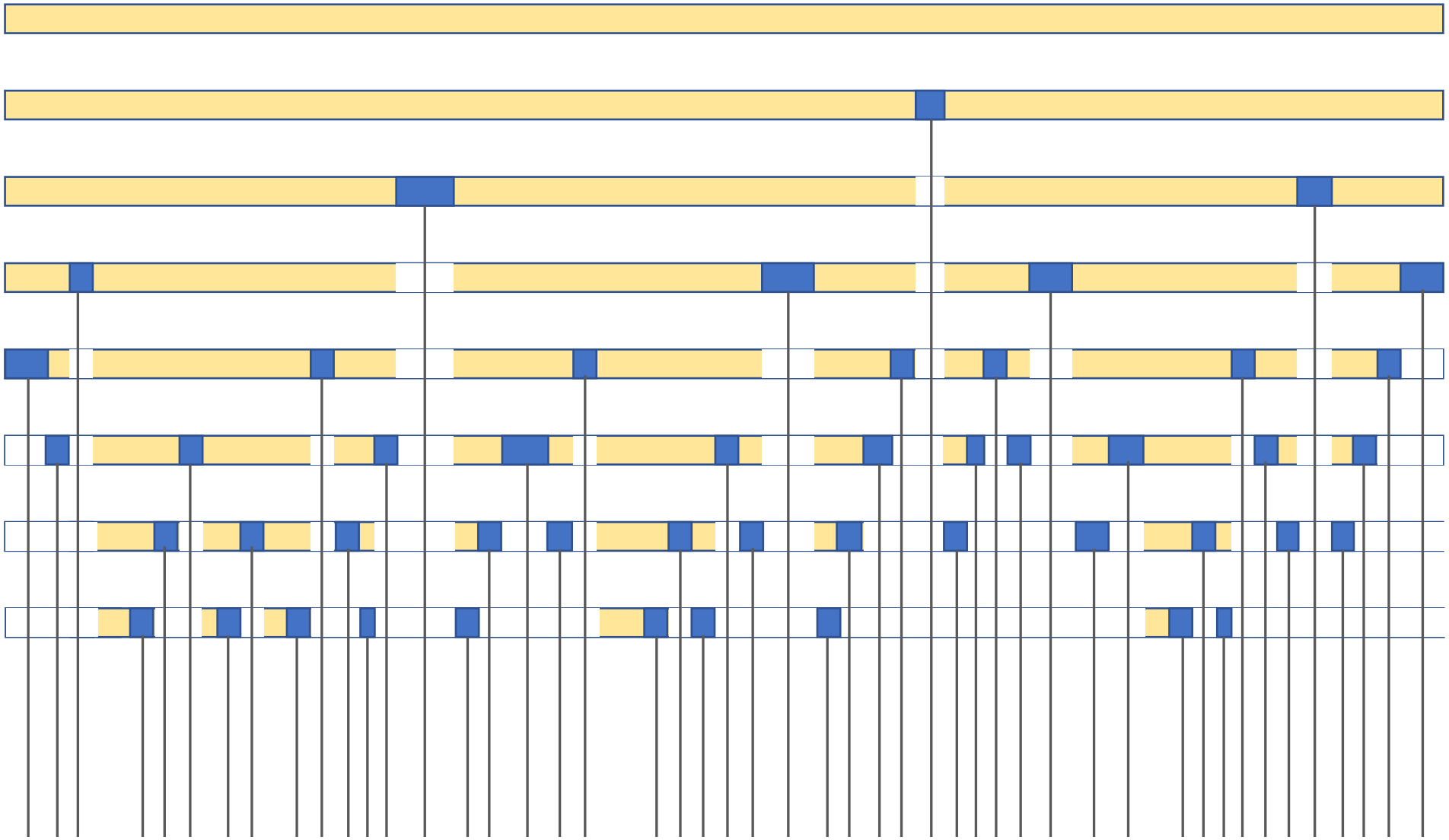


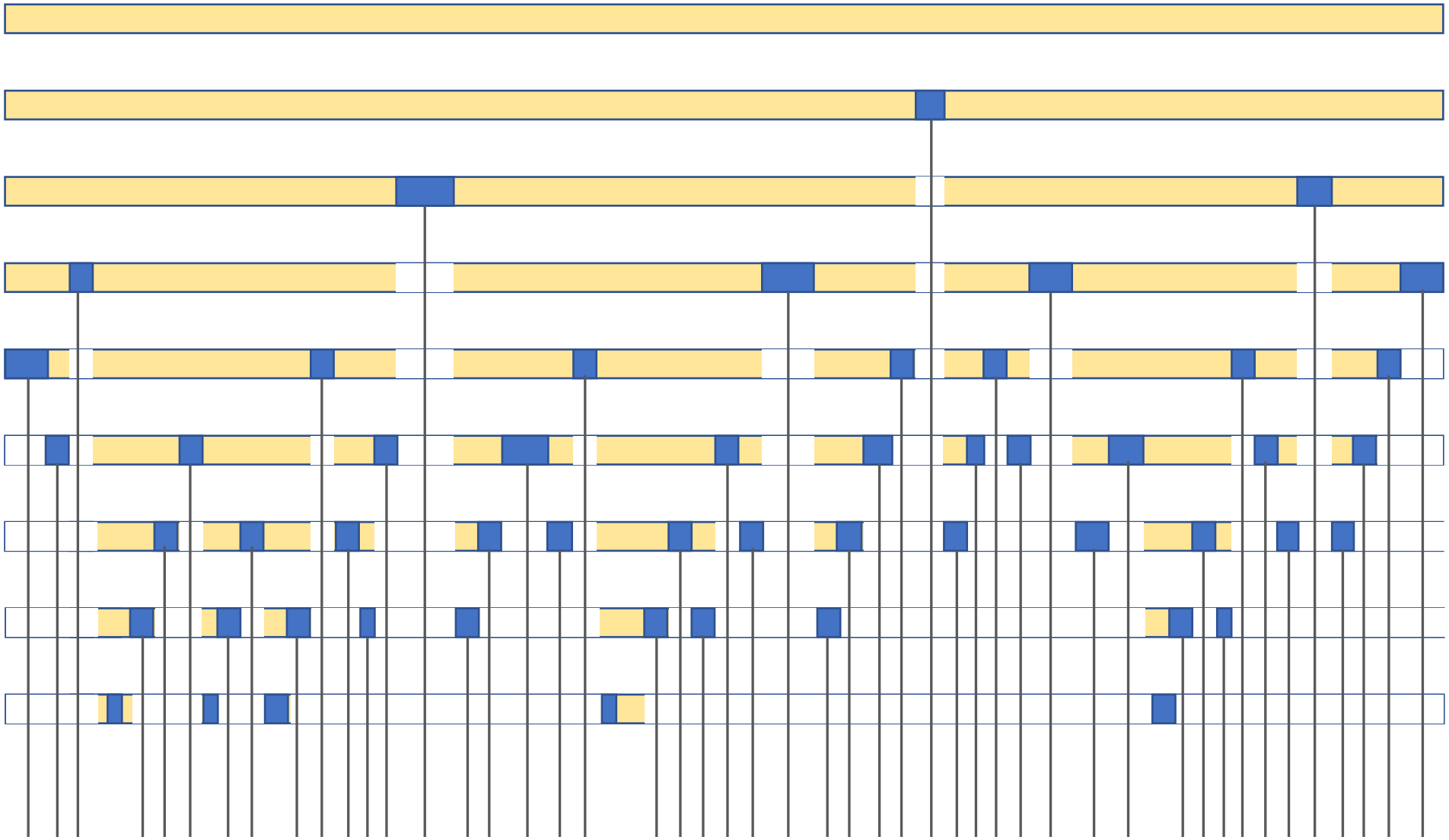


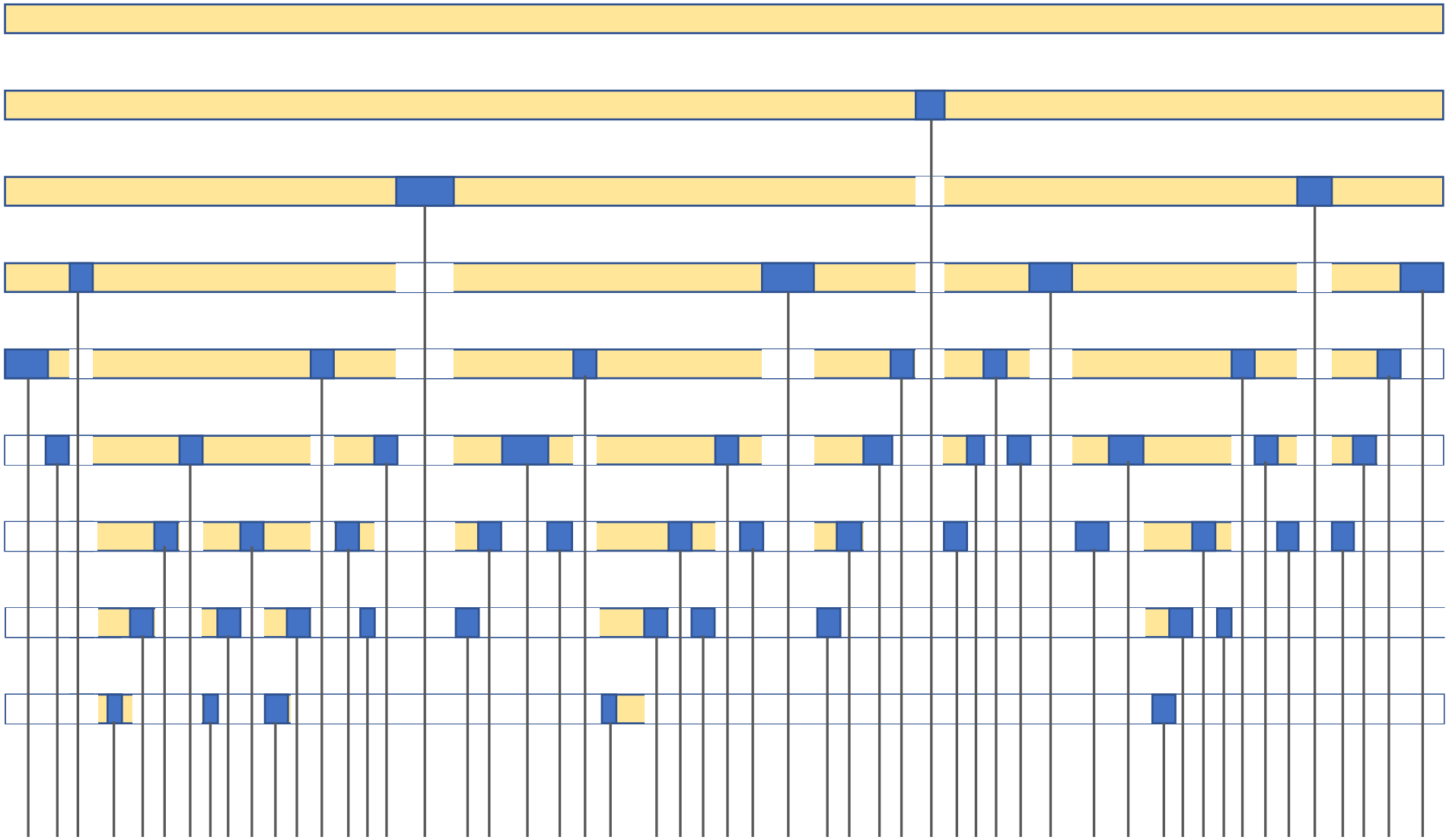


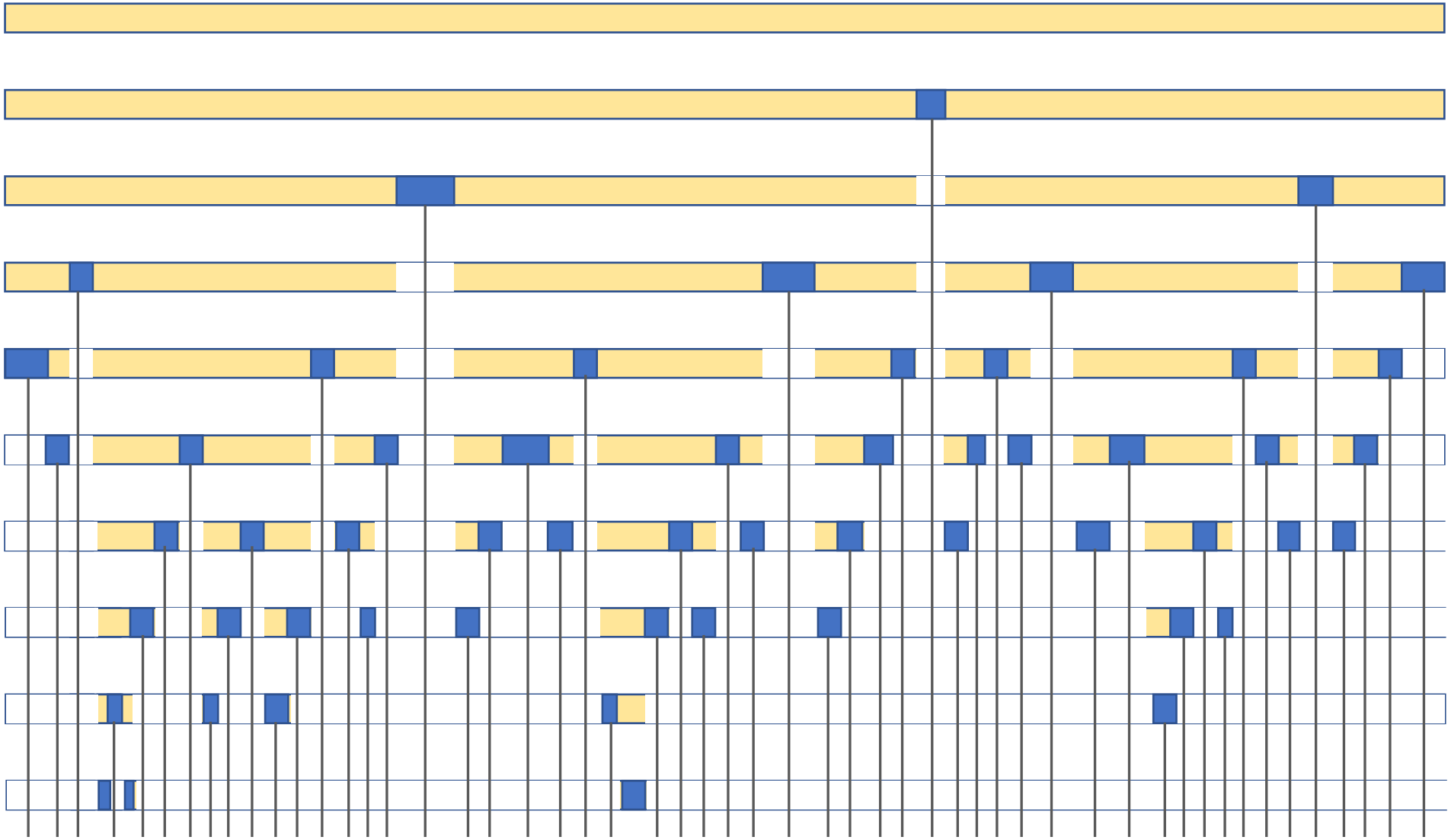


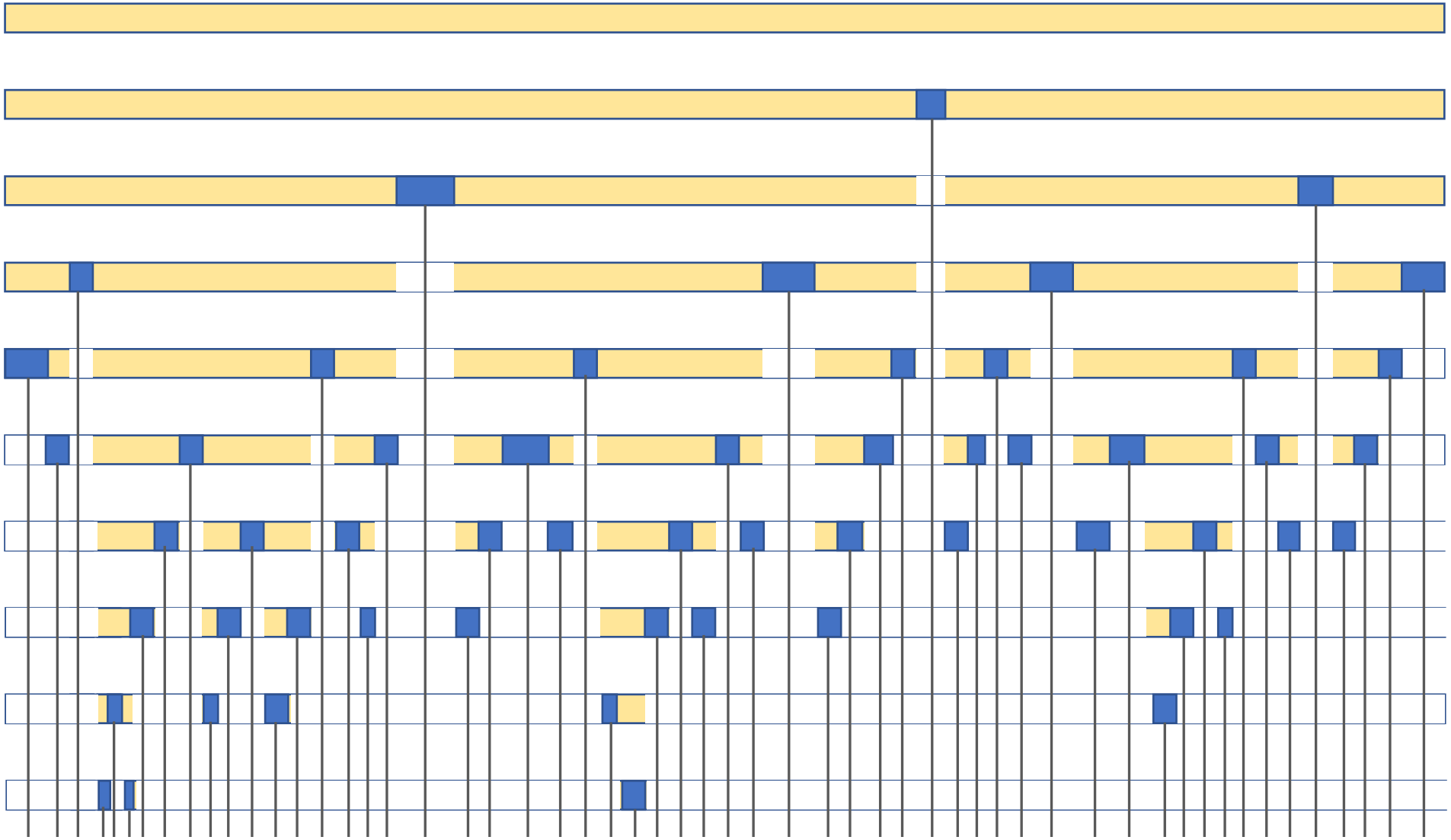


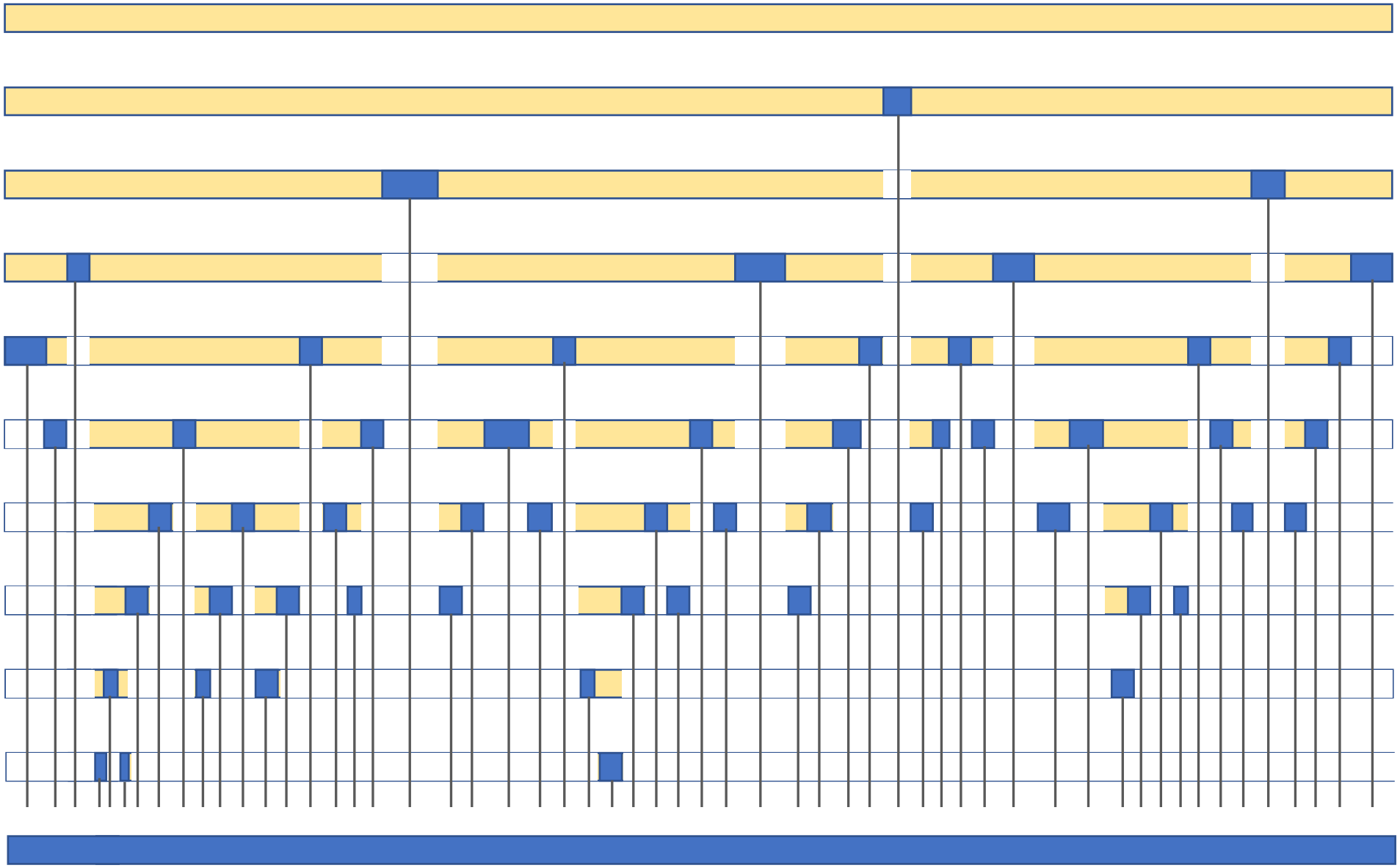


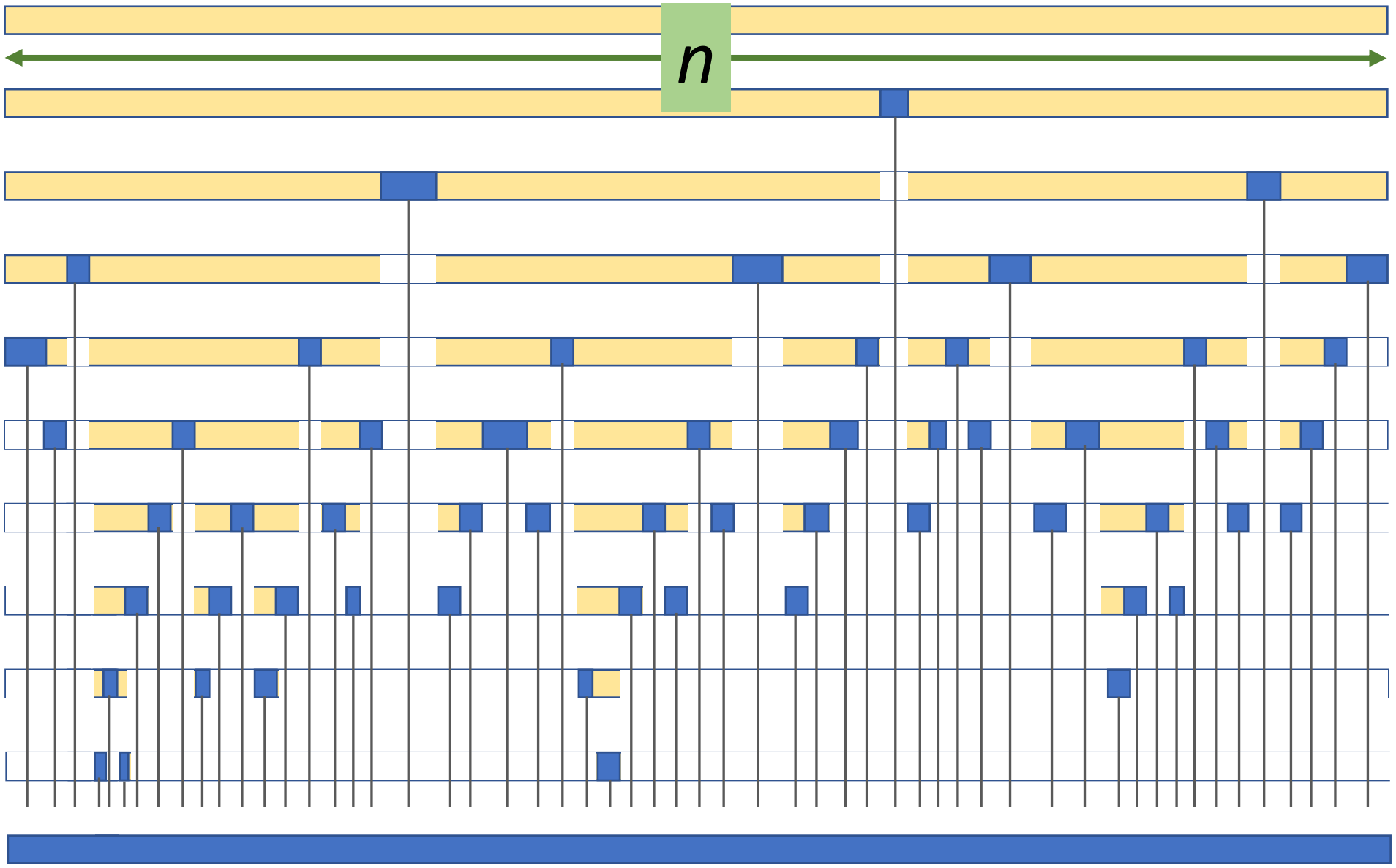


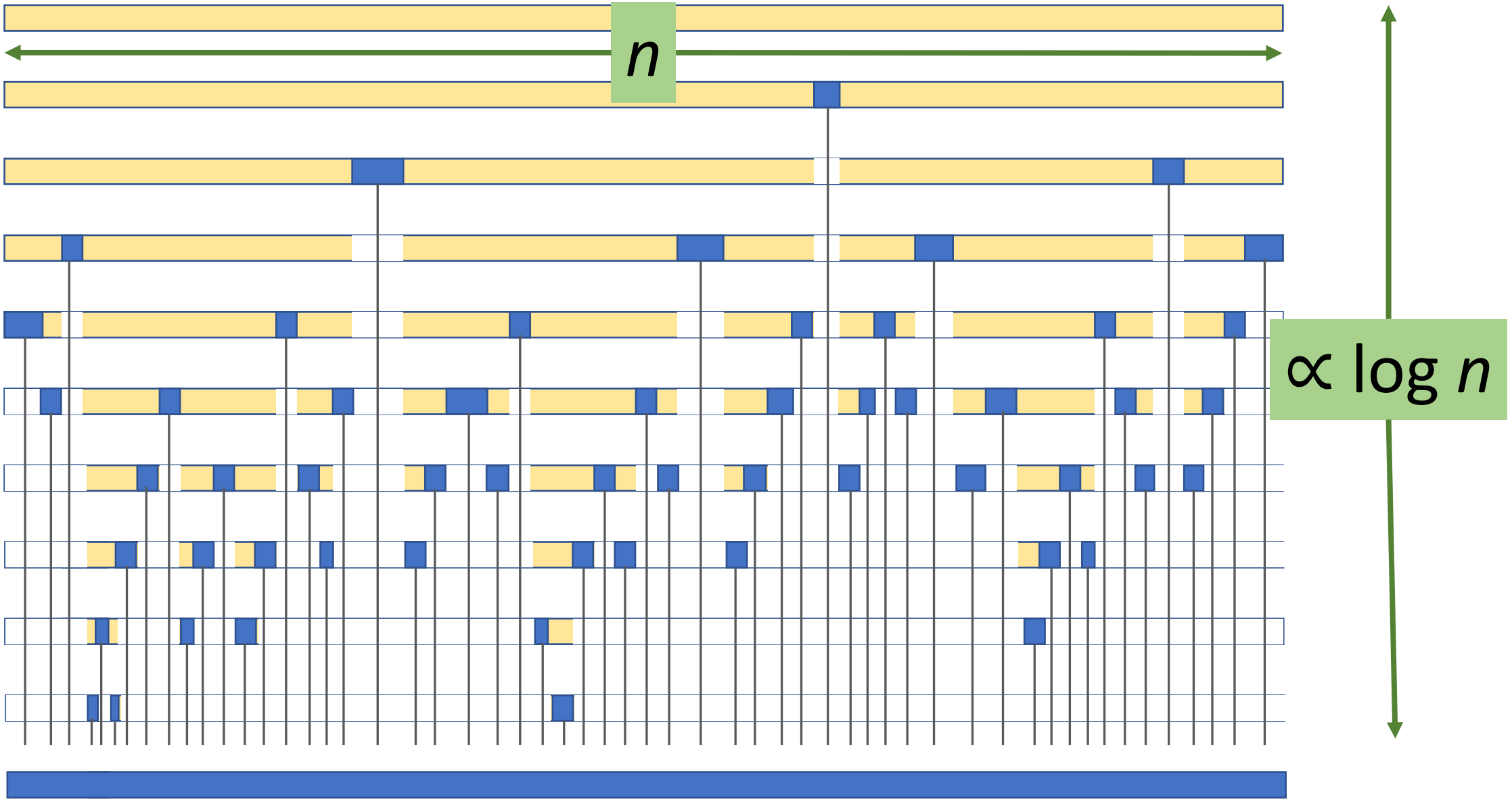









n


n
∝ log n