[320] Search order and Queue Structures

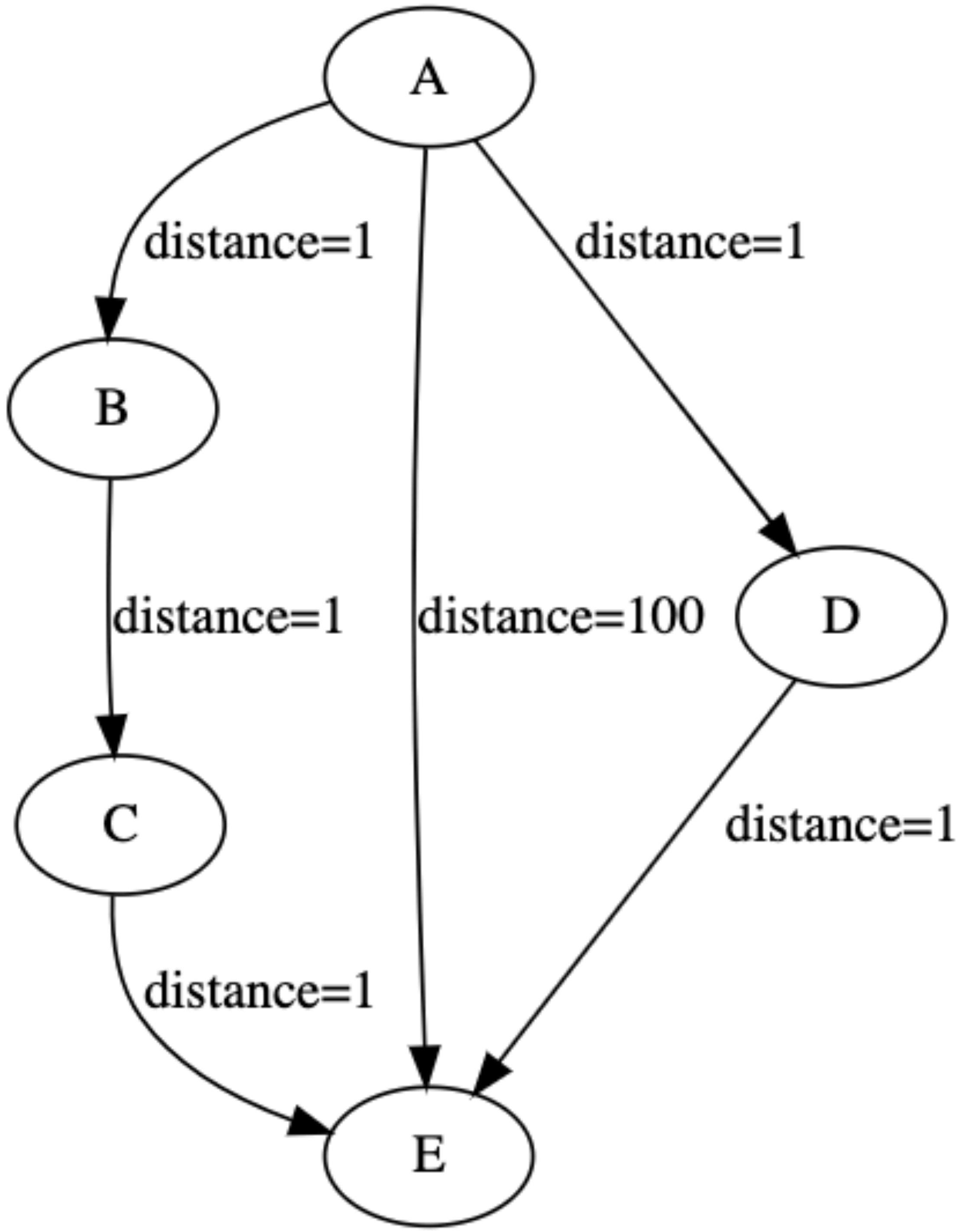
Tyler Caraza-Harter

Shortest Weighted Path

What path will DFS choose?

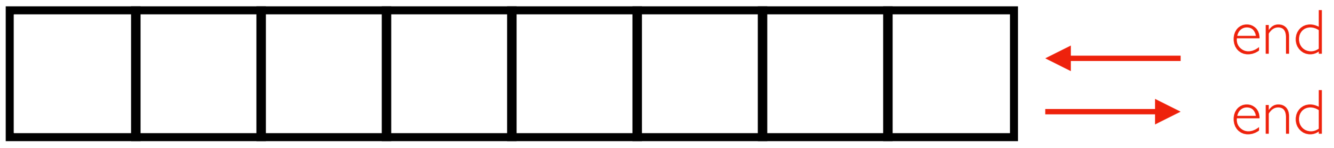
What path will BFS choose?

What path would you choose?

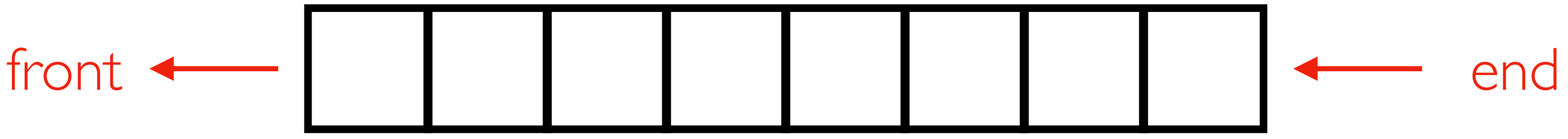


Your "to do" list

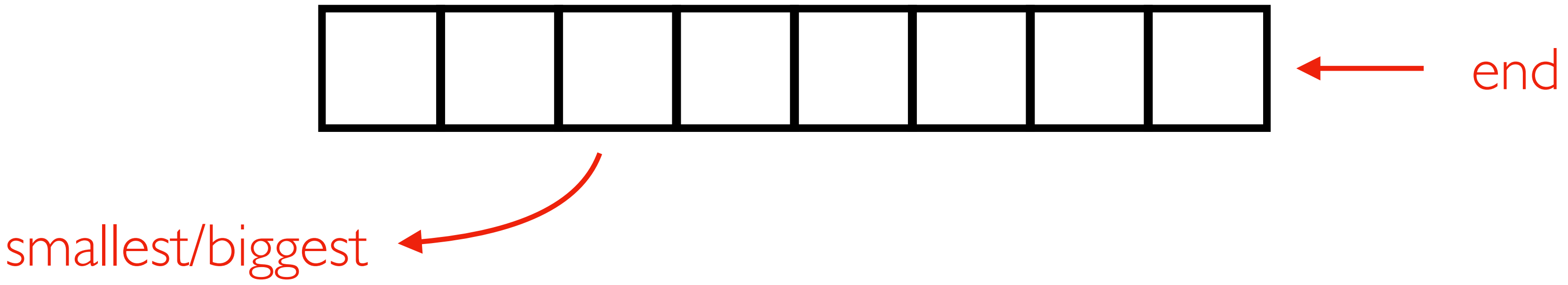
Stack



Queue



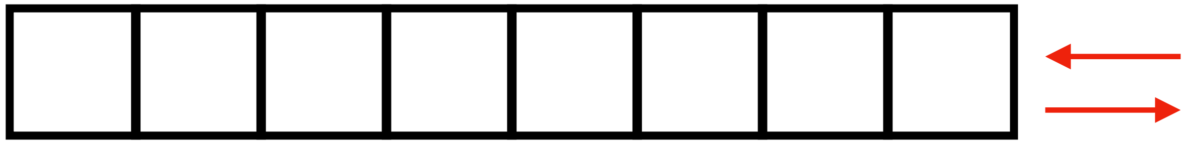
Priority Queue



Your "to do" list

Stack

```
L.append(x)
```



end

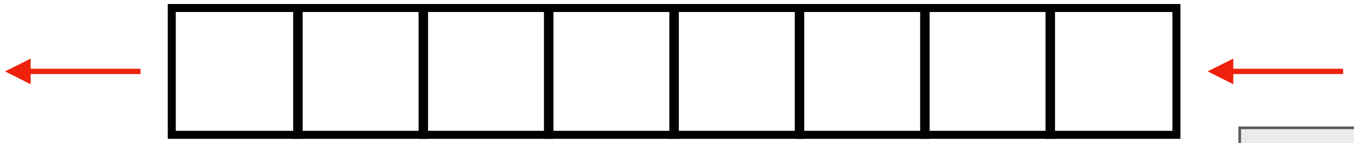
end

```
x = L.pop(-1)
```

Queue

```
x = L.pop(0)
```

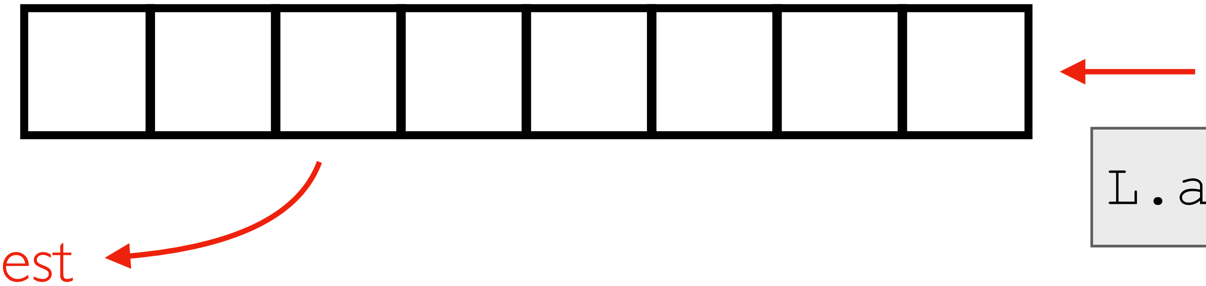
front



end

```
L.append(x)
```

Priority Queue



end

```
L.append(x)
```

smallest/biggest

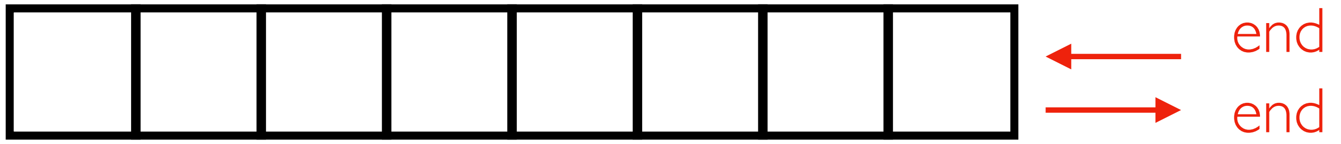
```
L.sort()
x = L.pop(-1)
```

what operations are slow?

Your "to do" list

```
L.append(x)
```

Stack

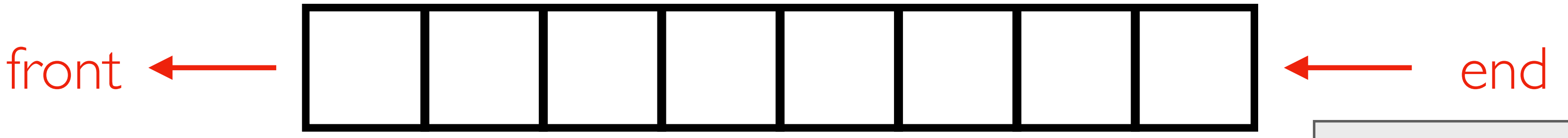


```
x = L.pop(-1)
```

!

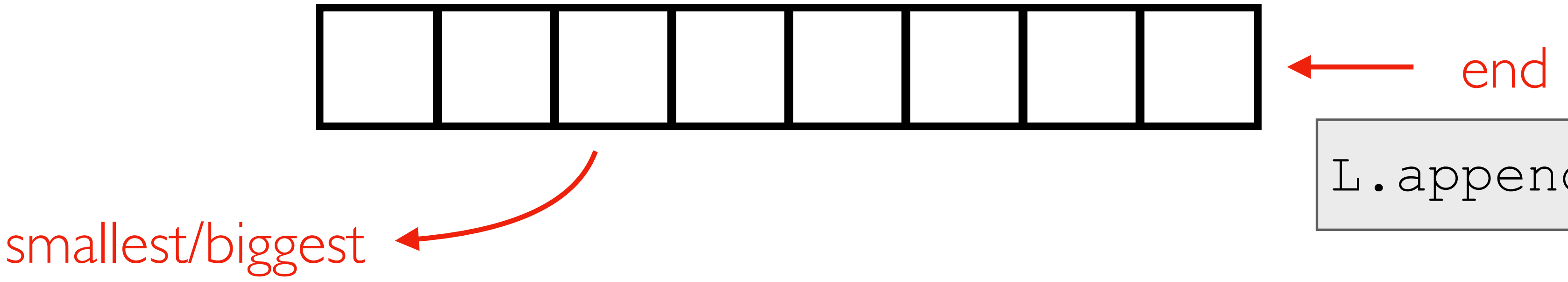
```
x = L.pop(0)
```

Queue



```
L.append(x)
```

Priority Queue



```
L.append(x)
```

!

```
L.sort()
x = L.pop(-1)
```

what operations are slow?

Complexity: Time vs. Memory

```
def ratio_search(L, target):
    for n in L:
        for d in L:
            if n/d == target:
                return True
    return False

def list_ratios(L):
    ratios = []
    for n in L:
        for d in L:
            ratios.append(n/d)
    return ratios
```

if N is len(L) and f(N) is the **number of steps**, with is the Big-O complexity of each function?

Complexity: Time vs. Memory

```
def ratio_search(L, target):
    for n in L:
        for d in L:
            if n/d == target:
                return True
    return False

def list_ratios(L):
    ratios = []
    for n in L:
        for d in L:
            ratios.append(n/d)
    return ratios
```

if N is len(L) and f(N) is the **max memory** used, with is the Big-O complexity of each function?

Review: Stacks, Queues, Priority Queues

1

```
while len(todo):
    curr = todo.pop(0)

    # other code...
    # appends to todo
```

2

```
while len(todo):
    todo.sort()
    curr = todo.pop(0)

    # other code...
    # appends to todo
```

3

```
while len(todo):
    curr = todo.pop(-1)

    # other code...
    # appends to todo
```

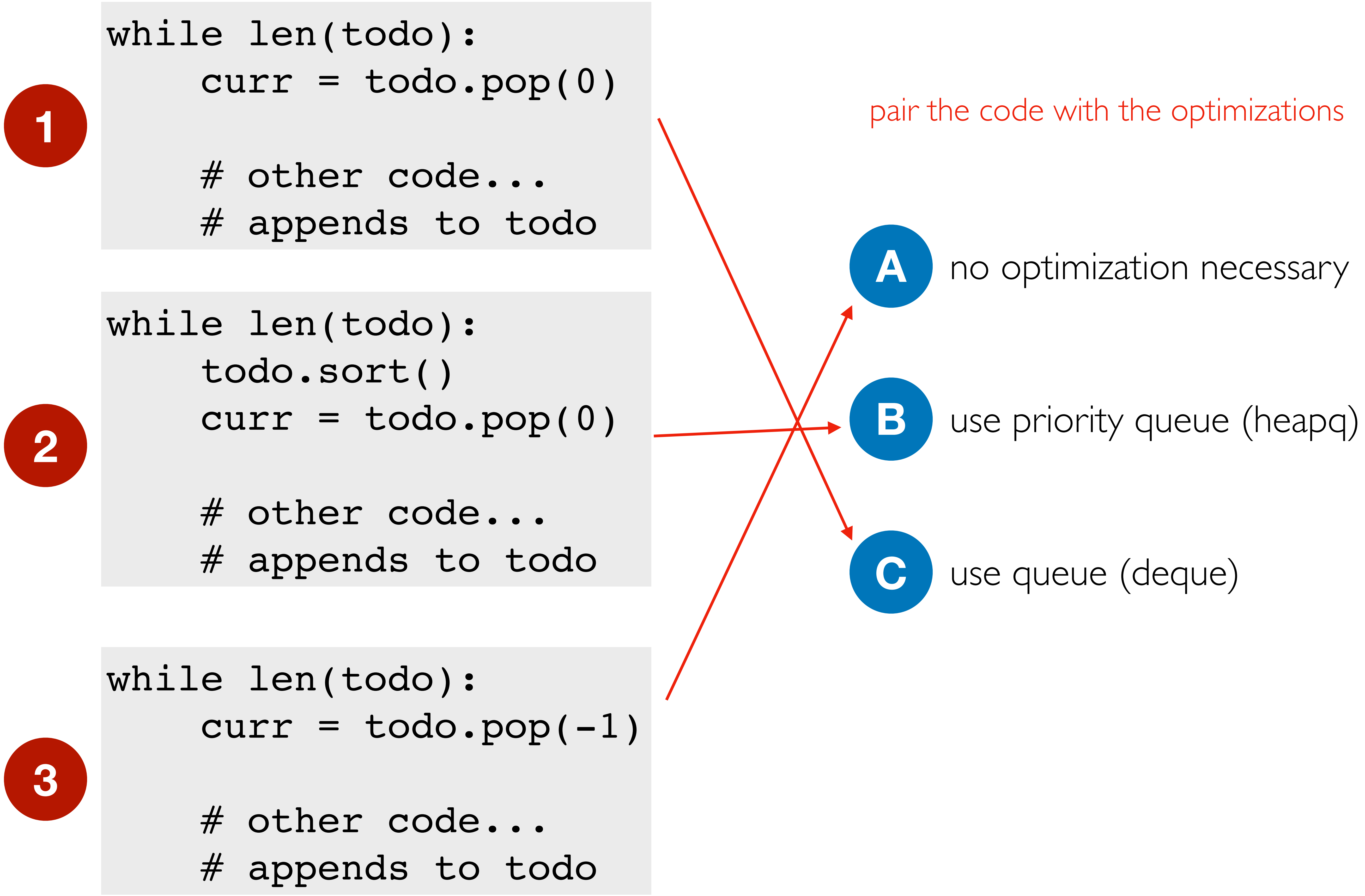
pair the code with the optimizations

A no optimization necessary

B use priority queue (heapq)

C use queue (deque)

Review: Stacks, Queues, Priority Queues



1

```
while len(todo):
    curr = todo.pop(0)

    # other code...
    # appends to todo
```

2

```
while len(todo):
    todo.sort()
    curr = todo.pop(0)

    # other code...
    # appends to todo
```

3

```
while len(todo):
    curr = todo.pop(-1)

    # other code...
    # appends to todo
```

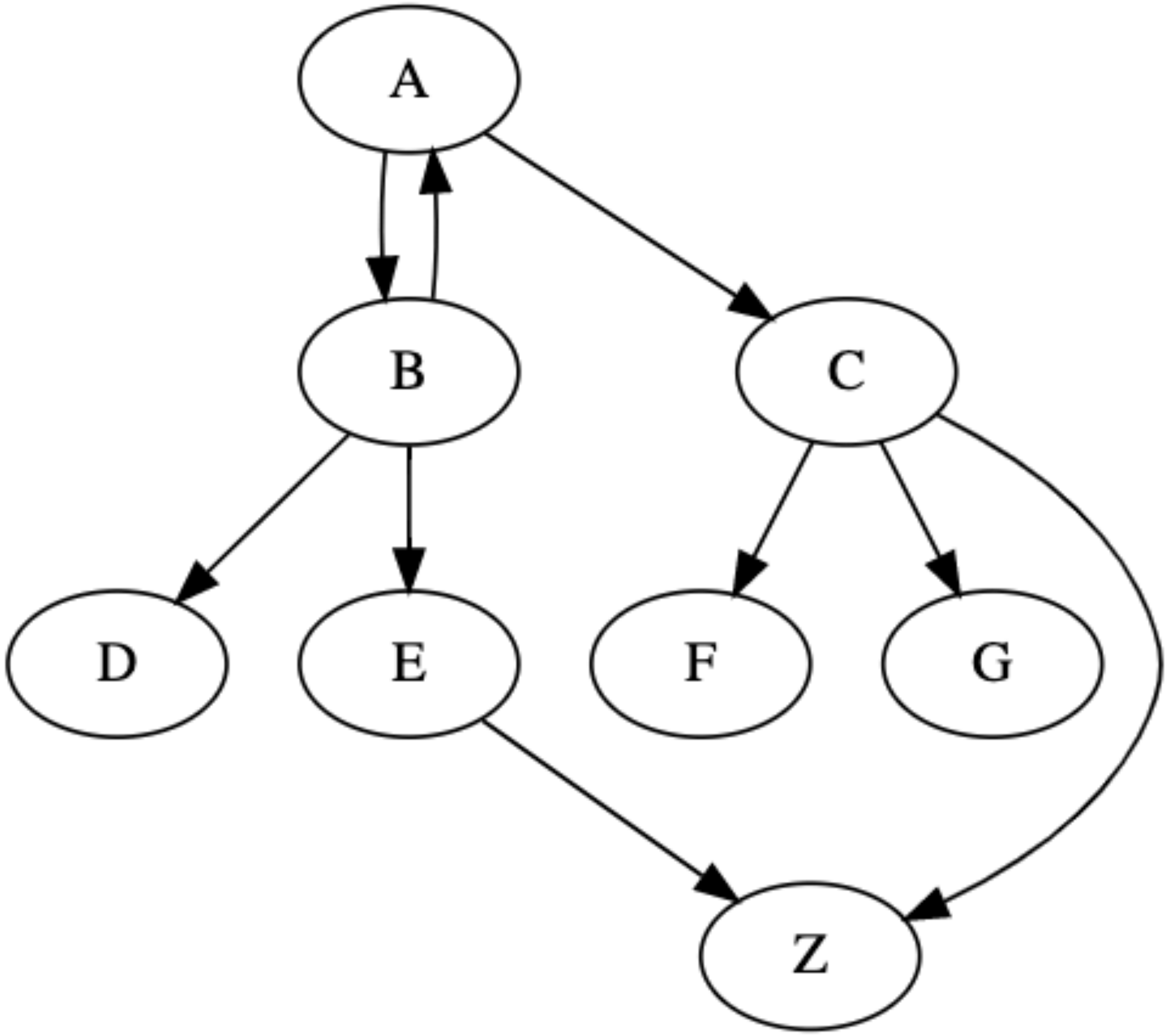
pair the code with the optimizations

A no optimization necessary

B use priority queue (heapq)

C use queue (deque)

Review: Search Order



How many grandchildren does A have?

Assume any loop over a node's edges goes left to right.

We want to find an A-to-Z path.

With DFS:

- what path is found?
- what is the traversal order?

With BFS:

- what path is found?
- what is the traversal order?