Lecture 15

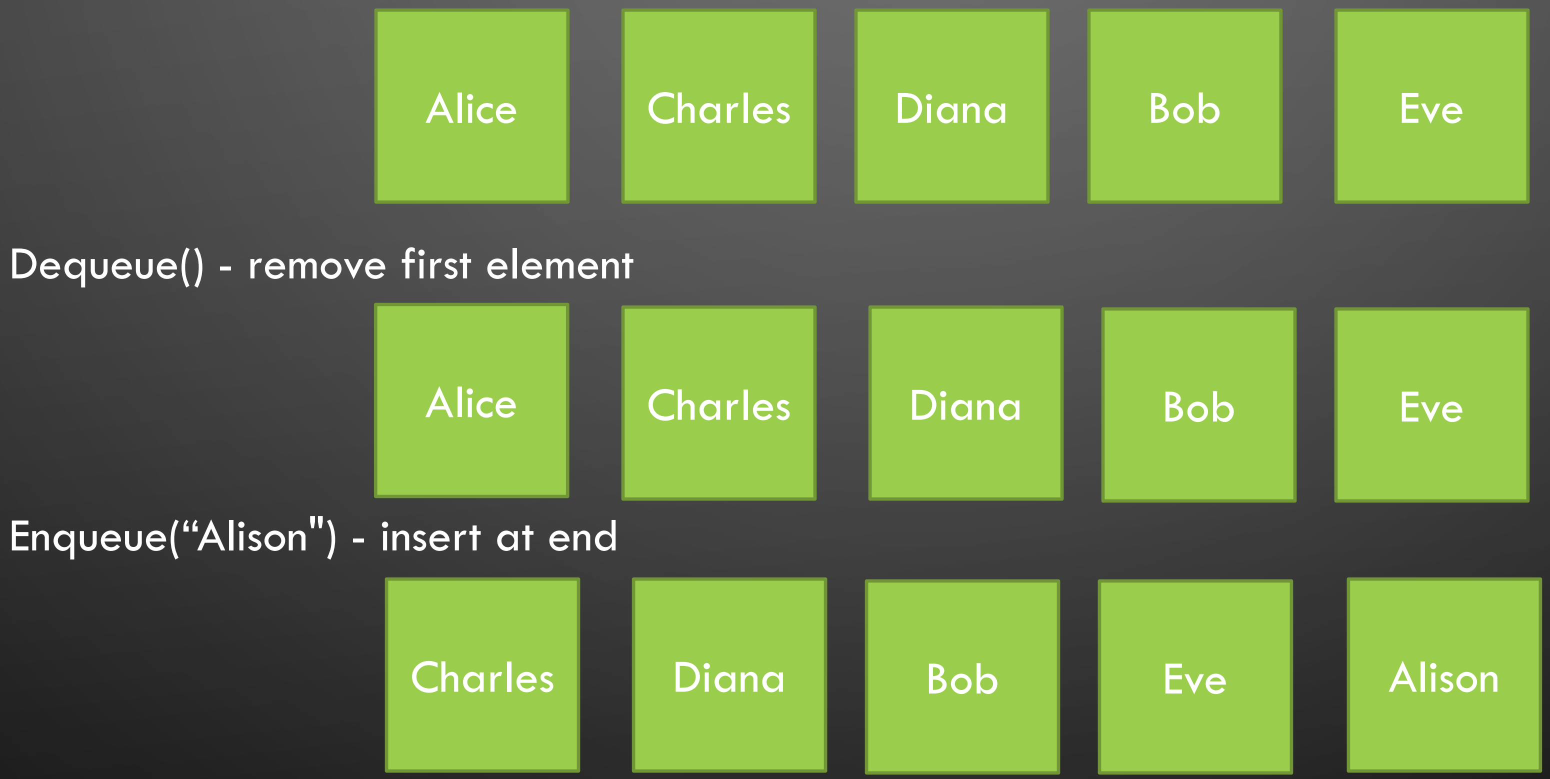
QUEUE AND DEQUEUE ABSTRACT DATA TYPE

Outline

- What is the Queue ADT?
- Single-Ended Queue
- Double-Ended Queue

Queue: First-in-First-Out (FIFO)

- Real-life examples
  - Line for a rollercoaster, line at the store
  - Structure to order bus or network communications



Alice Charles Diana Bob Eve

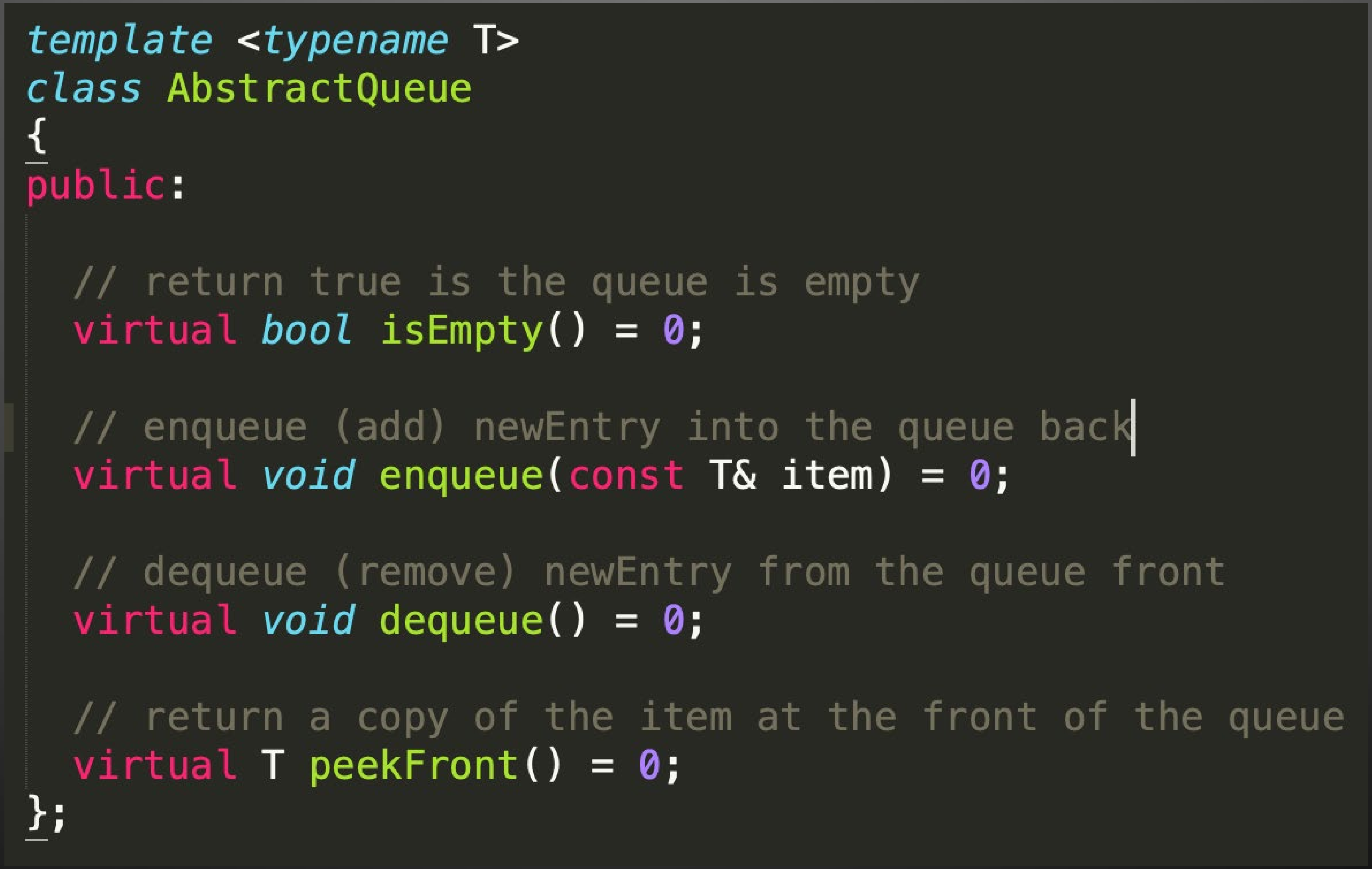
Dequeue() - remove first element

Alice Charles Diana Bob Eve

Enqueue(“Alison") - insert at end

Charles Diana Bob Eve Alison

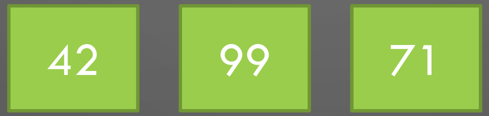
The Queue ADT



```
template <typename T>
class AbstractQueue
{
public:

  // return true is the queue is empty
  virtual bool isEmpty() = 0;

  // enqueue (add) newEntry into the queue back
  virtual void enqueue(const T& item) = 0;

  // dequeue (remove) newEntry from the queue front
  virtual void dequeue() = 0;

  // return a copy of the item at the front of the queue
  virtual T peekFront() = 0;
};
```

Example: Contents of Queue for each step

1. q.enqueue(42)
2. q.enqueue(99)
3. q.enqueue(87)
4. q.dequeue()
5. q.enqueue(71)
6. q.dequeue()
7. q.dequeue()
8. q.dequeue()



42 99 71

Using an Array for a Single-Ended Queue

- enqueue() inserts in the array position N+1
- dequeue() removes from the array at position 0
- PeekFront returns getEntry(0)

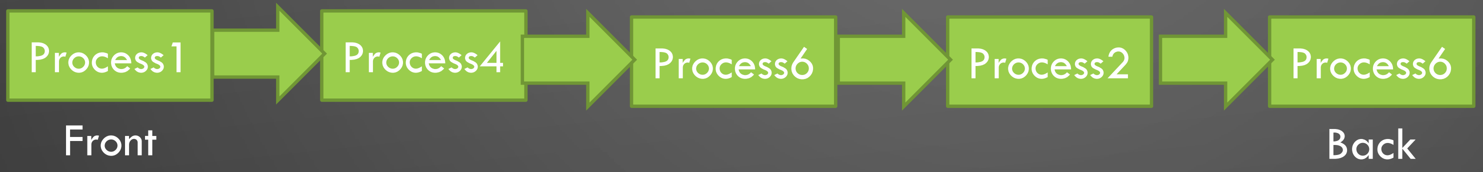
0	1	2	3	4	5	6	7
Process 1	Process 4	Process 6	Process 2	Process 6	Process 4	Process 1	Process 9

Example: Operating System Scheduling

- What is the complexity of enqueue and dequeue?
- Can also enqueue at position 0 and dequeue at position N
  - How does this decision impact order of complexity of enqueue and dequeue?

Using a LinkedList for a Single-Ended Queue

- enqueue inserts in the back of the linked list
- dequeue removes the first element in the linked list
- PeekFront returns getEntry(0)



Process1 Process4 Process6 Process2 Process6

Front

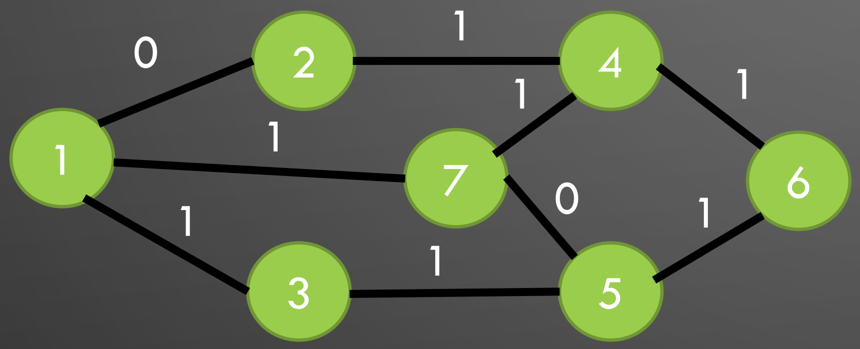
Back

Example: Operating System Scheduling

- How do the complexity of enqueue and dequeue compare to an array-based implementation?

Double-Ended Queue

- A queue where you can add/remove from either side
  - Practical application example: more efficient search for shortest path in a *binary weighted graph*



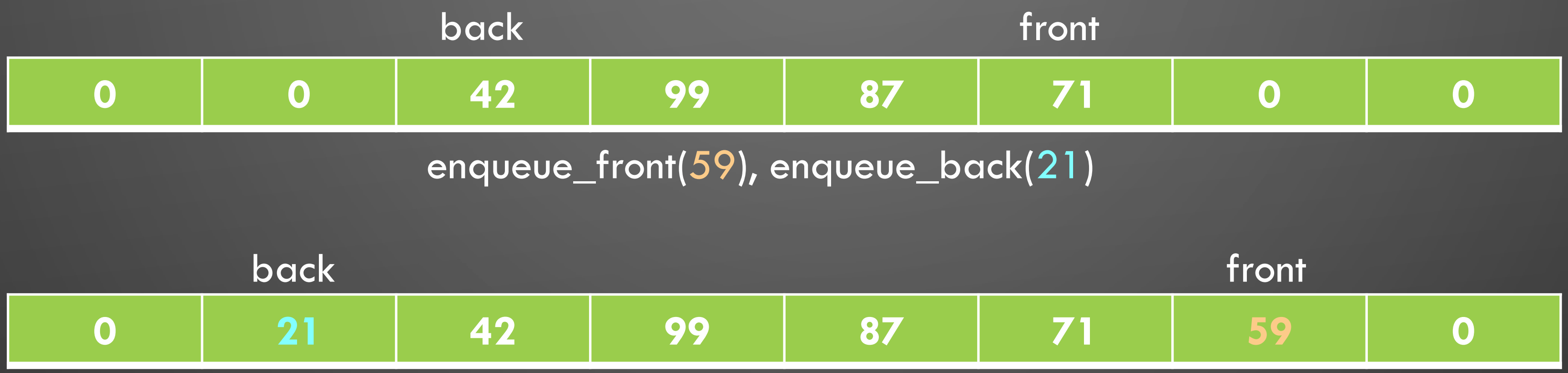
Simple binary weighted graph:

1) The weights between nodes are the cost to get from one node to another
2) Some nodes cannot directly connect to each other without going through another node
3) It is a binary weighted graph because all weights are either 0 or 1

While computing the distance from each node to each other node, ‘0’ weight edges are added to the front, while ‘1’ weight edges are added to the back

Double-Ended Queue: Ring Buffer

- Fixed-size, array-based, double-ended queue
- Prevents adding more than the fixed size



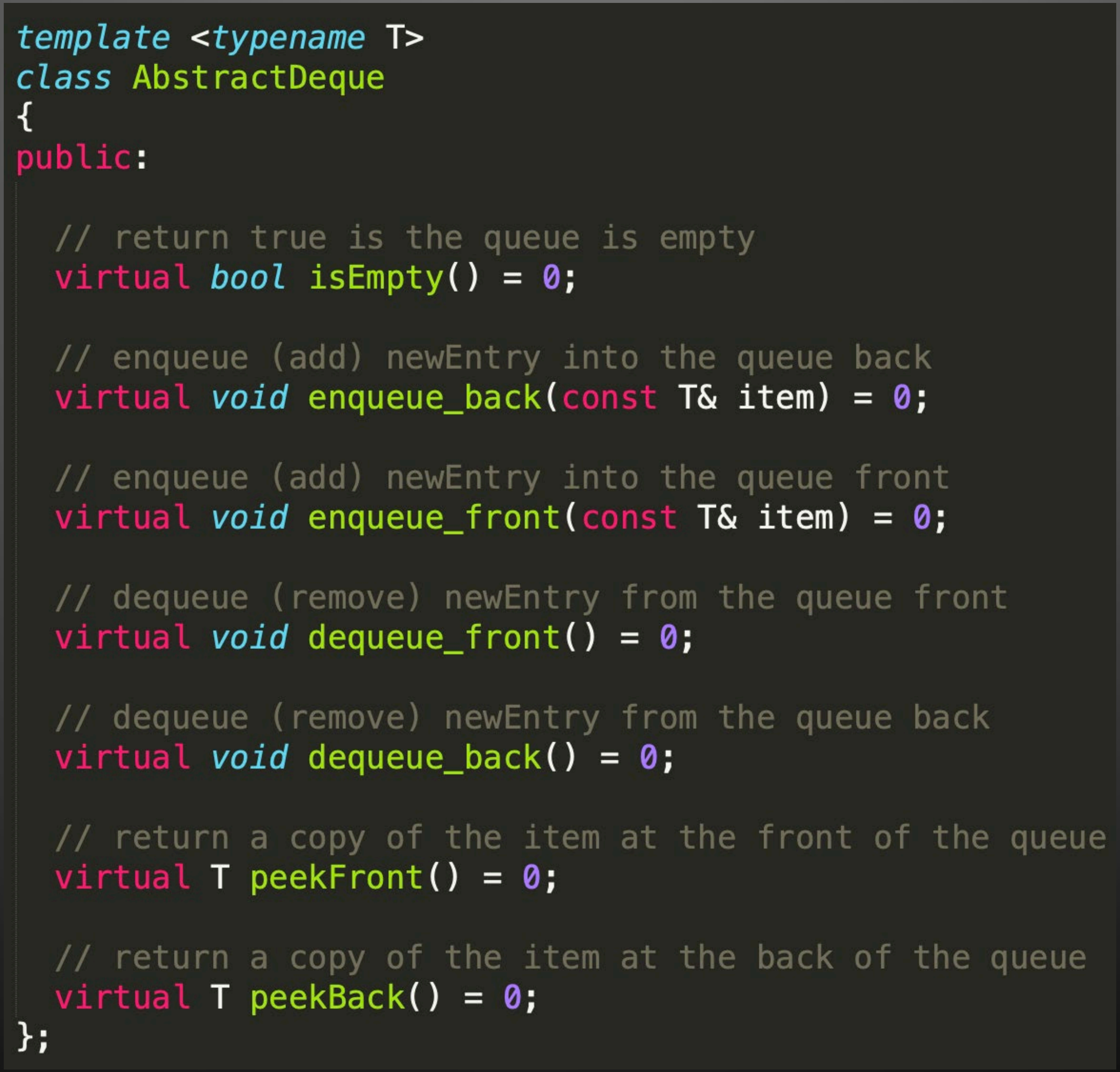
		back			front		
0	0	42	99	87	71	0	0

enqueue_front(59), enqueue_back(21)

	back					front	
0	21	42	99	87	71	59	0

- How do the enqueue and dequeue compare in computational complexity to the non-fixed size double-ended queue?

Double-Ended Queue



```
template <typename T>
class AbstractDeque
{
public:

  // return true is the queue is empty
  virtual bool isEmpty() = 0;

  // enqueue (add) newEntry into the queue back
  virtual void enqueue_back(const T& item) = 0;

  // enqueue (add) newEntry into the queue front
  virtual void enqueue_front(const T& item) = 0;

  // dequeue (remove) newEntry from the queue front
  virtual void dequeue_front() = 0;

  // dequeue (remove) newEntry from the queue back
  virtual void dequeue_back() = 0;

  // return a copy of the item at the front of the queue
  virtual T peekFront() = 0;

  // return a copy of the item at the back of the queue
  virtual T peekBack() = 0;
};
```

Performance of array- & link-based implementations

operation	Array		Link	
	Best	Worst	Best	Worst
enqueue_front	O(1)	O(n)	O(1)	O(1)
dequeue_front	O(1)	O(n)	O(1)	O(1)
peekFront	O(1)	O(1)	O(1)	O(1)
enqueue_back	O(1)	O(n)	O(1)	O(1)
dequeue_back	O(1)	O(1)	O(1)	O(1)
peekBack	O(1)	O(1)	O(1)	O(1)

Assignment/Homework

- Reading pp. 399-410,431-432
- ICE 6 due on Tuesday.
- P3 not late Penalty if submitted by July 3rd (4th of July Promotion)
- ICE 4 Queue and Ring Buffer released.
- P4 PathFinder released