QUEUE

CSC212:Data Structure

Queue

- Queue: First In First Out (FIFO).
  - Used in operating systems, simulations etc.
- Priority Queues: Highest priority item is served first.
  - Used in operating systems, printer servers etc.

ADT Queue: Specification

Elements: The elements are of generic type <Type> (The elements are placed in nodes for linked list implementations).

Structure: the elements are linearly arranged, and ordered according to the order of arrival. Most recently arrived element is called the back or tail, and least recently arrived element is called the front or head.

Domain: the number of elements in the queue is bounded therefore the domain is finite. Type of elements: Queue

ADT Queue: Specification

Operations:

1. **Method** Enqueue (Type e)

   requires: Queue Q is not full. **input:** Type e.

   results: Element e is added to the queue at its tail. **output:** none.

2. **Method** Serve (Type e)

   requires: Queue Q is not empty. **input**: none

   results: the element at the head of Q is removed and its value assigned to e. **output**: Type e.

3. **Method** Length (int length)

   requires: none. **input**: none
   results: The number of element in the Queue Q is returned.
   output: length.

ADT Queue: Specification

Operations:

4. **Method** Full (boolean flag).

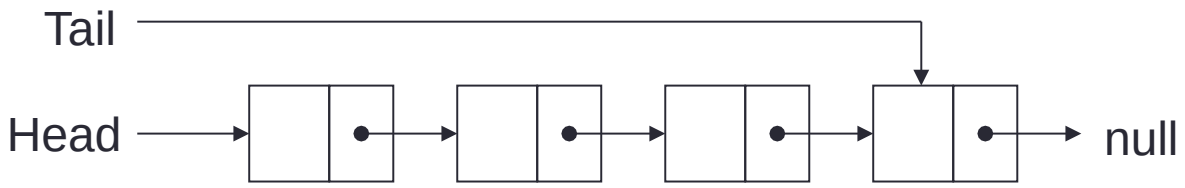
   requires: none. **input**: none

   results: If Q is full then flag is set to true, otherwise flag is set to false. **output**: flag.

Queue Interface

```
public interface Queue<T>{
public T serve( );
public void enqueue(T e);
public int length( );
public boolean full( );
}
```

ADT Queue (Linked-List)



ADT Queue (Linked-List): Element

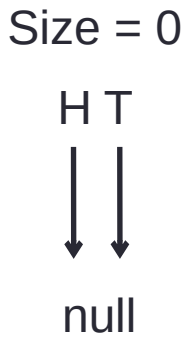
```
public class Node<T> {
  public T data;
  public Node<T> next;

  public Node() {
  data = null;
  next = null;
  }

  public Node(T val) {
  data = val;
  next = null;
  }

  // Setters/Getters?
}
```

ADT Queue (Linked-List): Representation

```
public class LinkedQueue<T> implements Queue<L>{
  private Node<T> head, tail;
  private int size;

  /** Creates a new instance of LinkedQueue */
  public LinkedQueue() {
      head = tail = null;
      size = 0;
  }
```

ADT Queue (Linked-List): Representation

```
public class LinkedQueue<T> {
  private Node<T> head, tail;
  private int size;

  /** Creates a new instance of LinkedQueue */
  public LinkedQueue() {
      head = tail = null;
      size = 0;
  }
```



ADT Queue (Linked-List): Implementation

```
public boolean full() {
    return false;
}

public int length (){
    return size;
}
```

ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 0

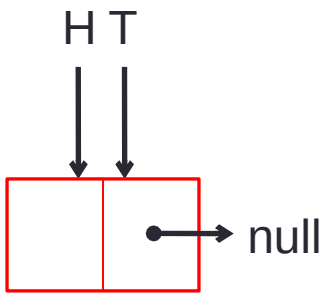
Example #1

ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 0

Example #1

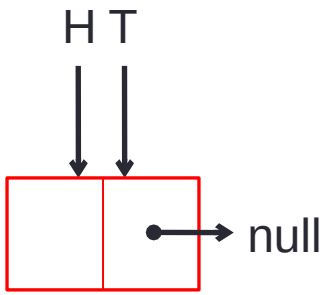


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 1

Example #1

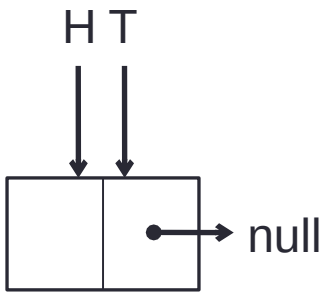


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 1

Example #2

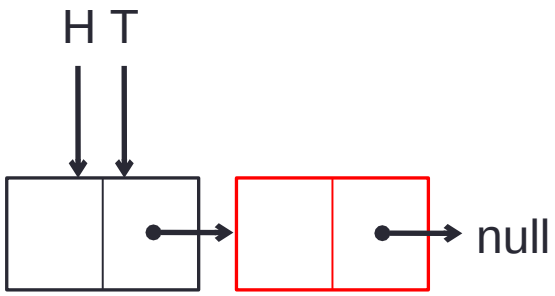


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 1

Example #2

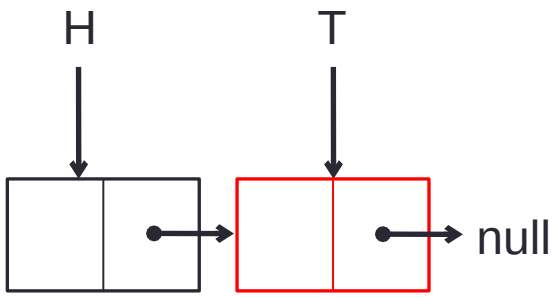


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 1

Example #2

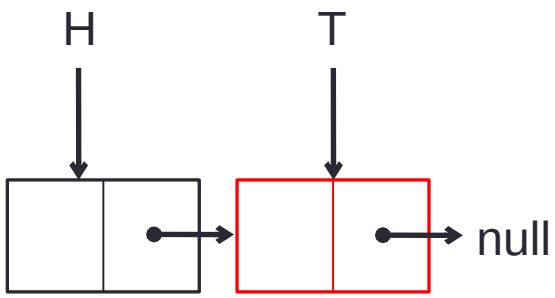


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 2

Example #2

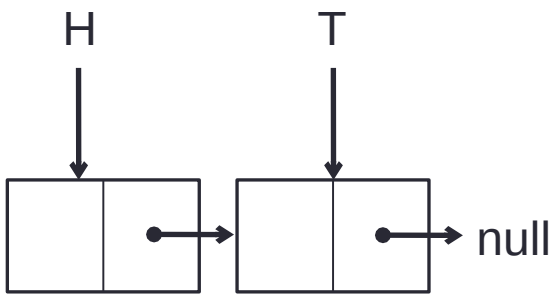


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 2

Example #3

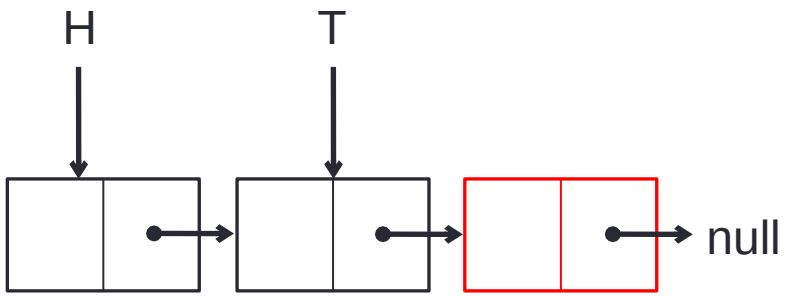


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 2

Example #3

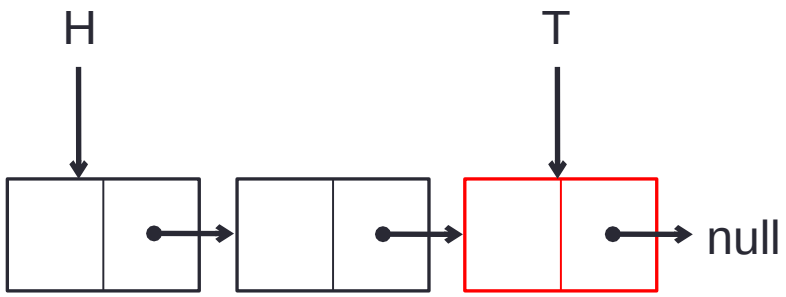


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 2

Example #3

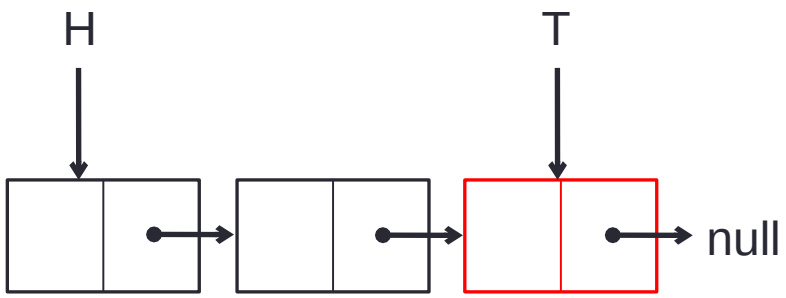


ADT Queue (Linked-List): Implementation

```
public void enqueue(T e) {
    if(tail == null){
        head = tail = new Node<T>(e);
    }
    else {
        tail.next = new Node<T>(e);
        tail = tail.next;
    }
    size++;
}
```

Size = 3

Example #3



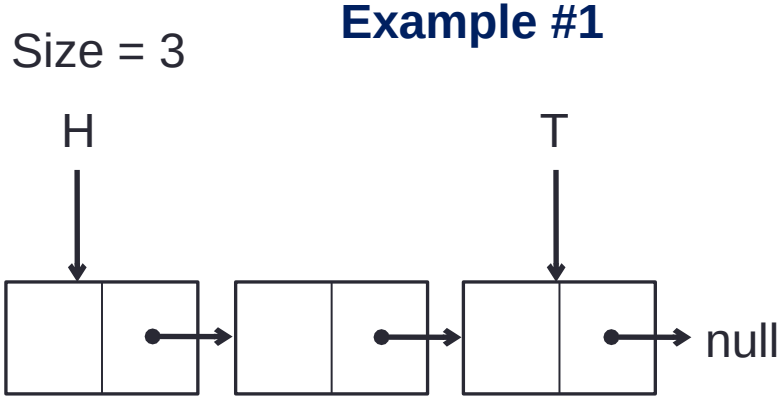
ADT Queue (Linked-List): Implementation

```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

ADT Queue (Linked-List): Implementation

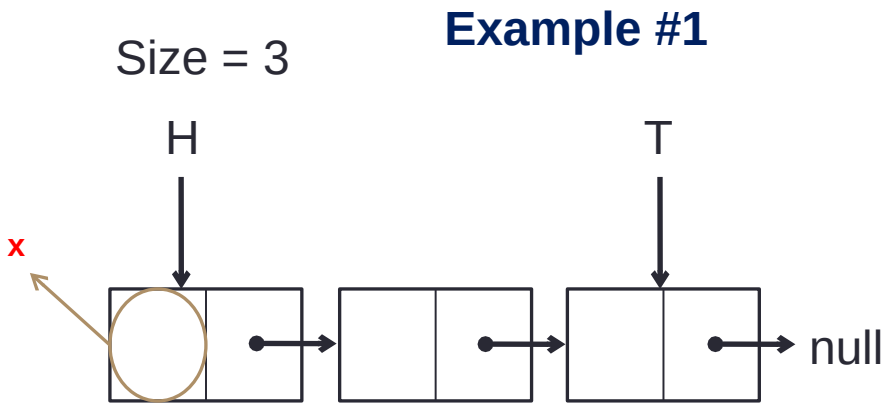
```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Example #1



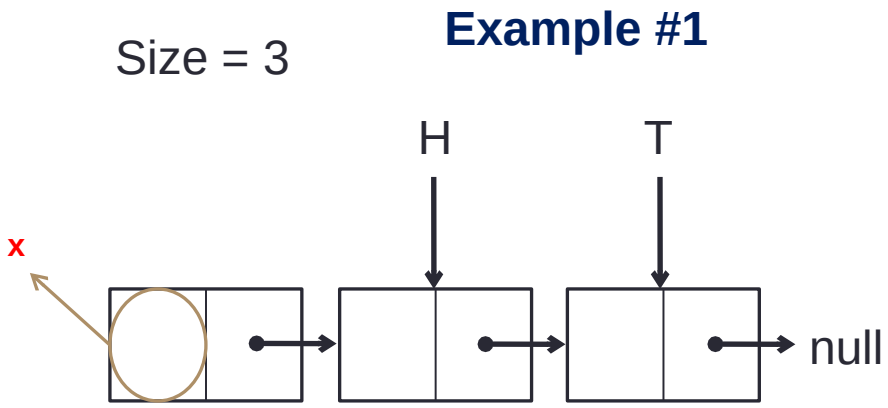
ADT Queue (Linked-List): Implementation

```
public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```



ADT Queue (Linked-List): Implementation

```
public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

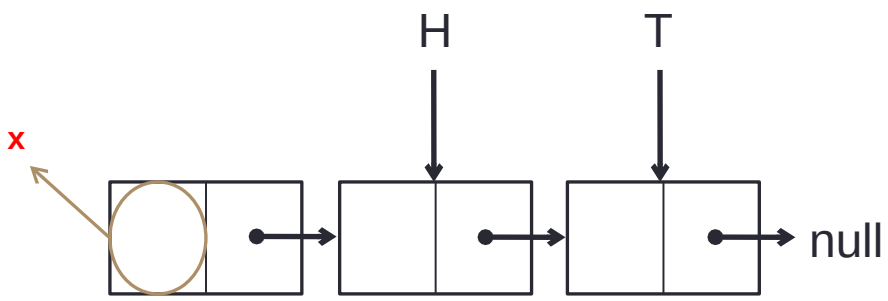


ADT Queue (Linked-List): Implementation

```
public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 2

Example #1

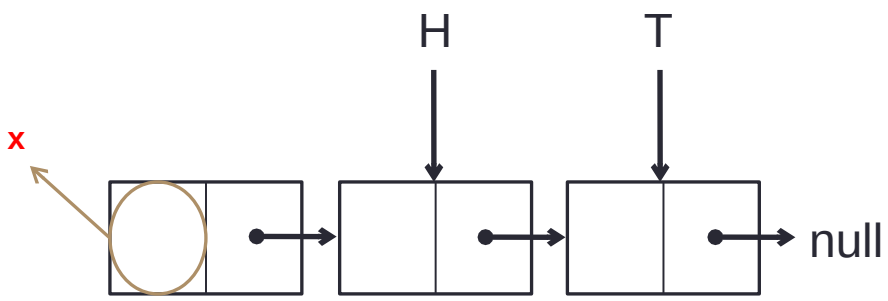


ADT Queue (Linked-List): Implementation

```
public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 2

Example #1

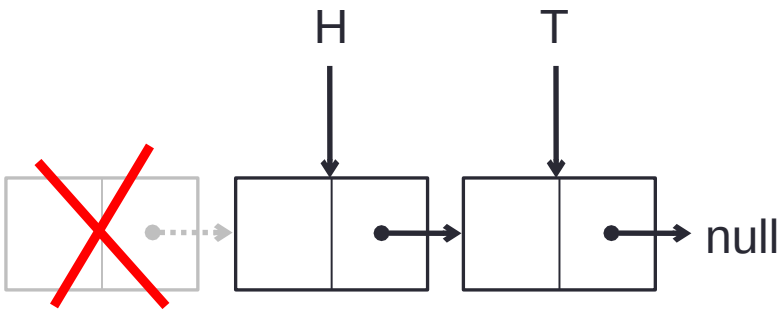


ADT Queue (Linked-List): Implementation

```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 2

Example #1

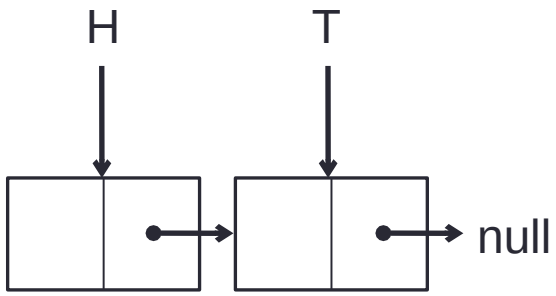


ADT Queue (Linked-List): Implementation

```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 2

Example #2

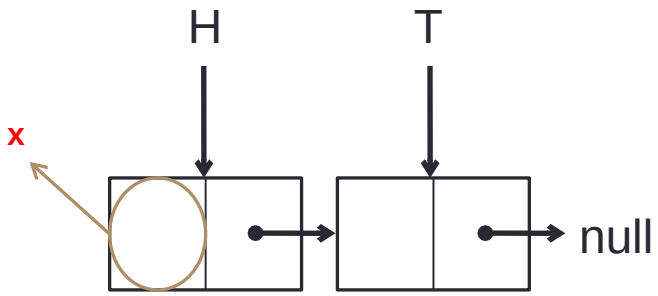


ADT Queue (Linked-List): Implementation

```
public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 2

Example #2

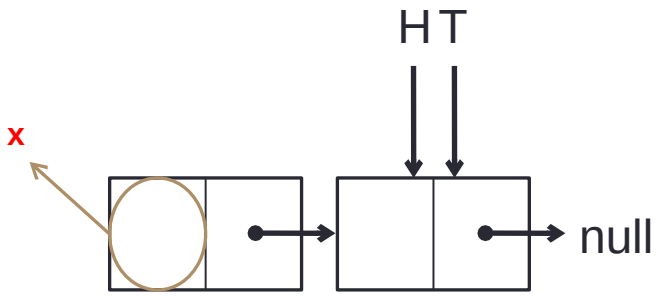


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 2

Example #2

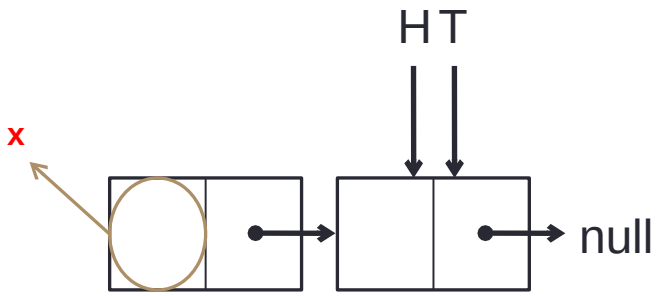


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 1

Example #2

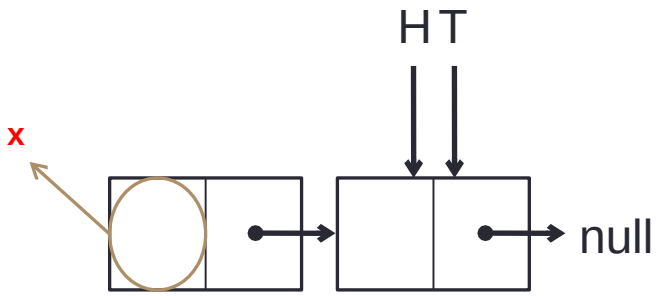


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 1

Example #2

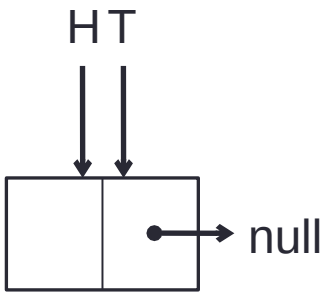


ADT Queue (Linked-List): Implementation

```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 1

Example #3

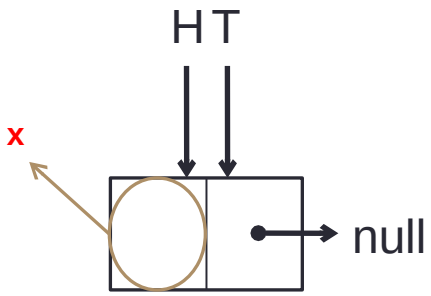


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 1

Example #3

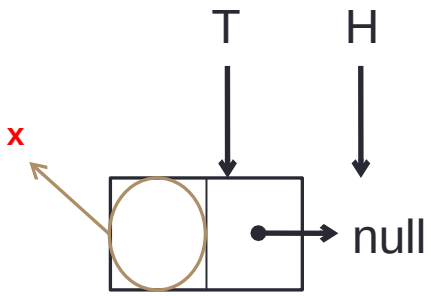


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 1

Example #3

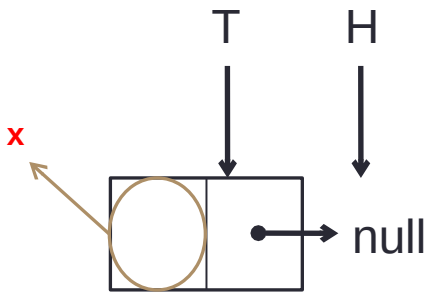


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 0

Example #3

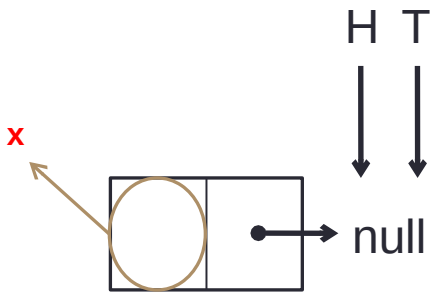


ADT Queue (Linked-List): Implementation

```
public T serve() {
      T x = head.data;
      head = head.next;
      size--;
      if(size == 0)
            tail = null;
      return x;
  }

}
```

Size = 0

Example #3

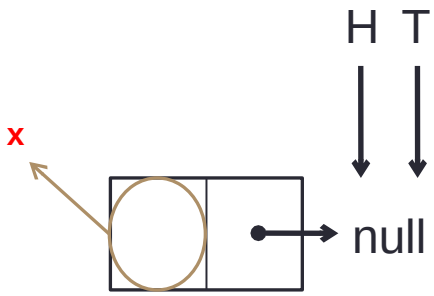


ADT Queue (Linked-List): Implementation

```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

Size = 0

Example #3



ADT Queue (Linked-List): Implementation

```
  public T serve() {
        T x = head.data;
        head = head.next;
        size--;
        if(size == 0)
              tail = null;
        return x;
  }

}
```

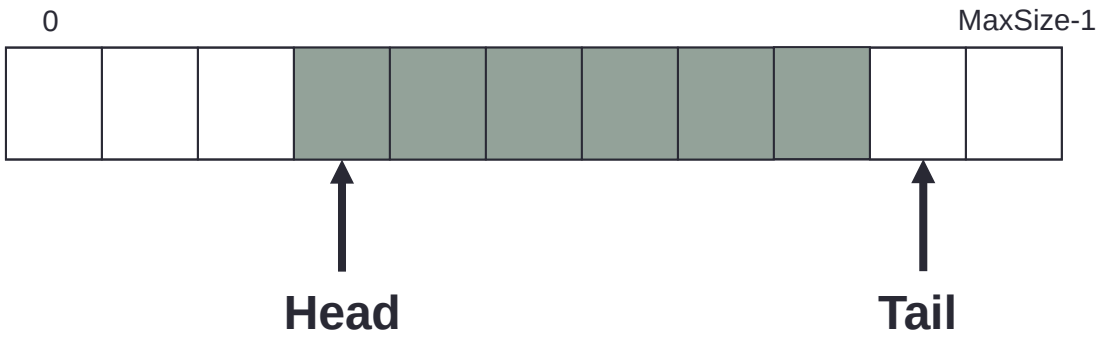
Size = 0

Example #3

ADT Queue (Array)

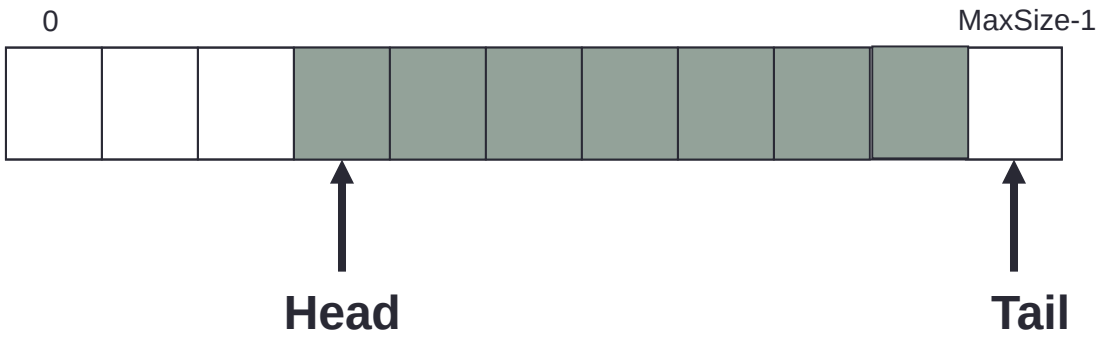
- A fixed size array is used to store the data elements.
- As data elements are enqueued & served the queue crawls through the array from low to high index values.
- As the queue crawls forward, it also expands and contracts.

ADT Queue (Array)



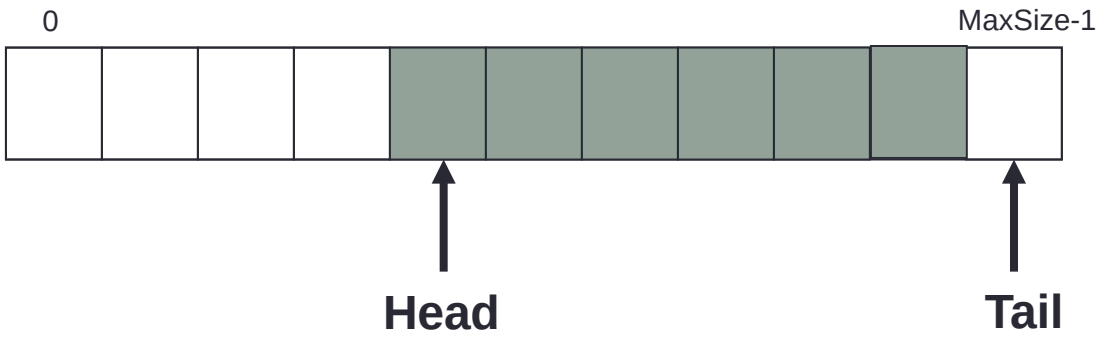
size=5

ADT Queue (Array)



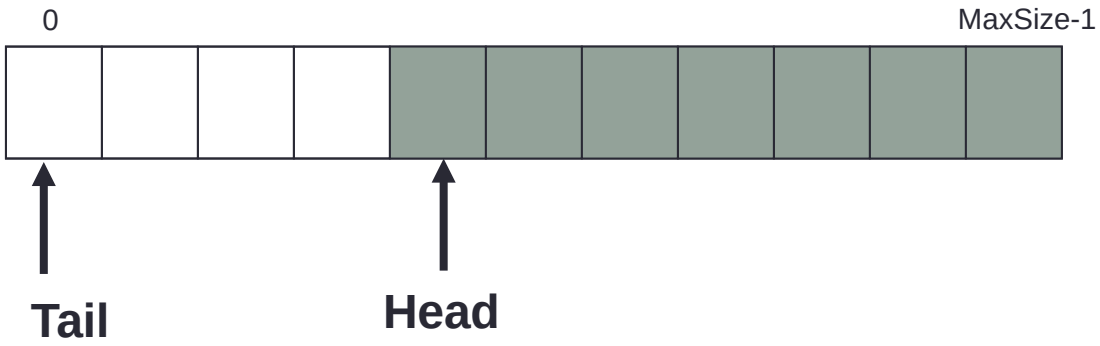
After one Enqueue

ADT Queue (Array)



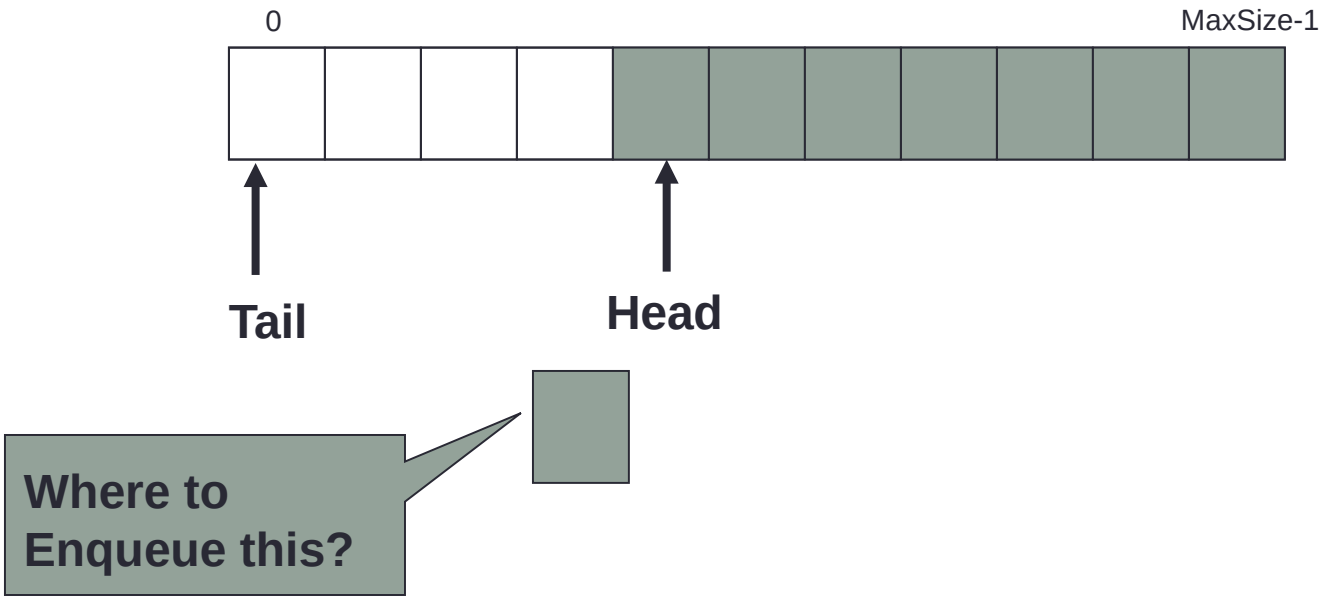
After one Serve

ADT Queue (Array)

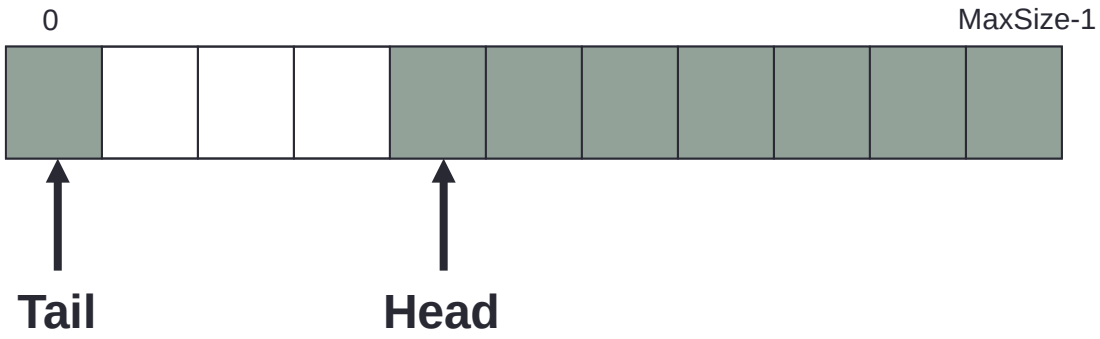


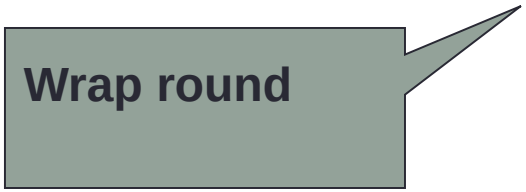
After one Enqueue

ADT Queue (Array)

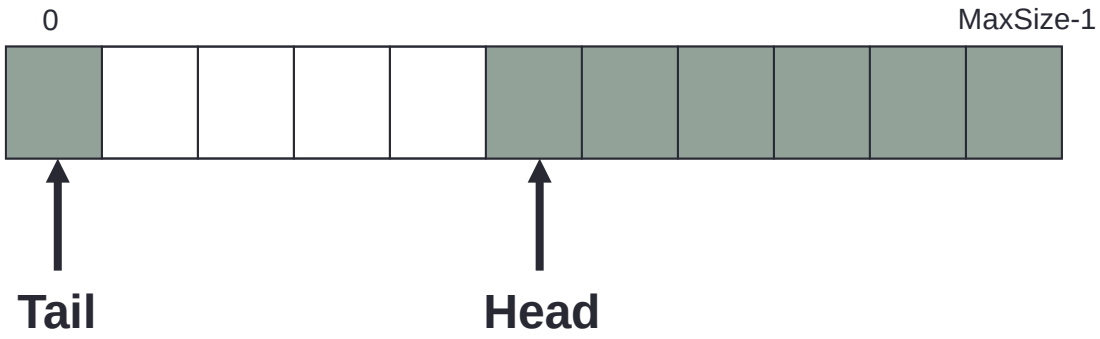


ADT Queue (Array)



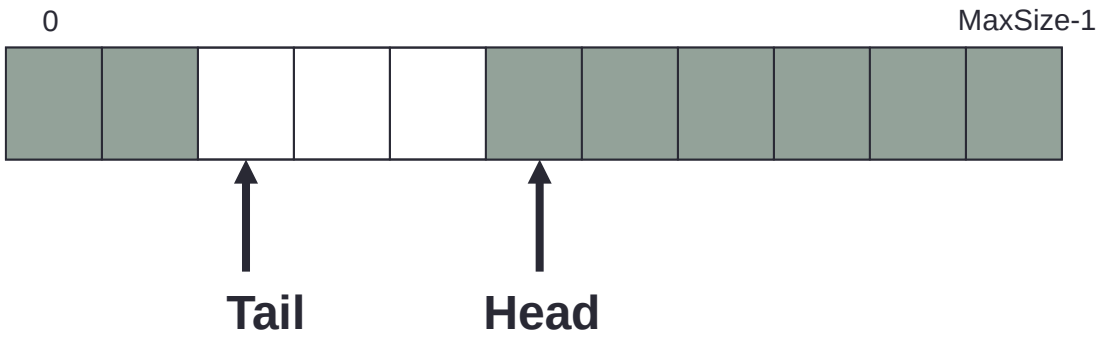


ADT Queue (Array)



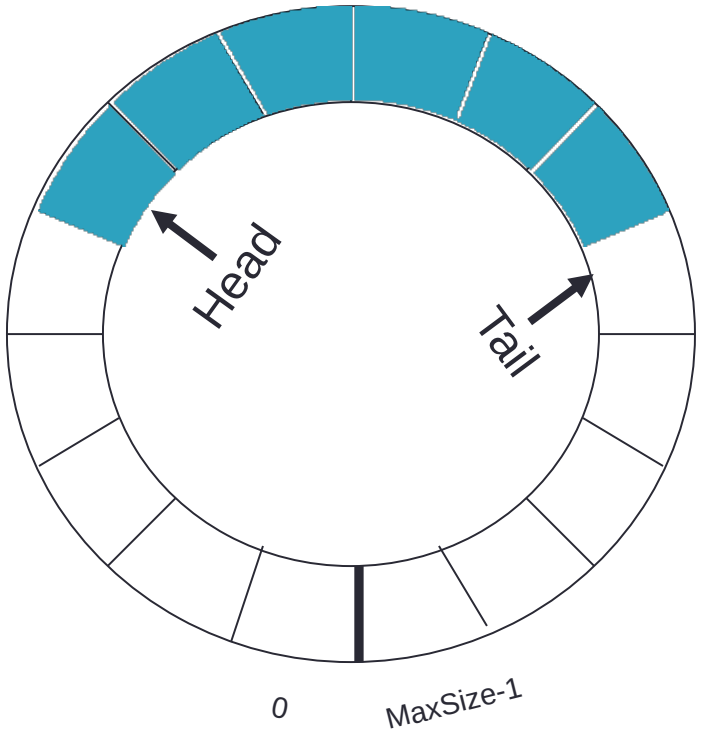
After one Serve

ADT Queue (Array)



After one Enqueue

ADT Queue (Array)

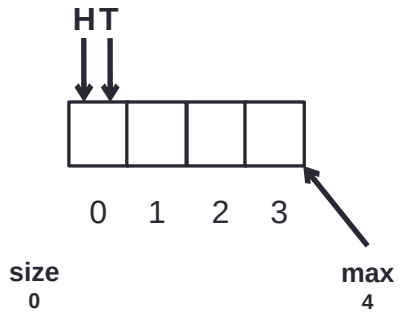


ADT Queue (Array): Representation

```
public class ArrayQueue<T> implements Queue<T>{
  private int maxsize;
  private int size;
  private int head, tail;
  private T[] nodes;

  /** Creates a new instance of ArrayQueue */
  public ArrayQueue(int n) {
        maxsize = n;
        size = 0;
        head = tail = 0;
        nodes = (T[])new Object[n];
  }
```

ADT Queue (Array): Representation

```
public class ArrayQueue<T> {
  private int maxsize;
  private int size;
  private int head, tail;
  private T[] nodes;

  /** Creates a new instance of ArrayQueue */
  public ArrayQueue(int n) {
      maxsize = n;
      size = 0;
      head = tail = 0;
      nodes = (T[])new Object[n];
  }
```

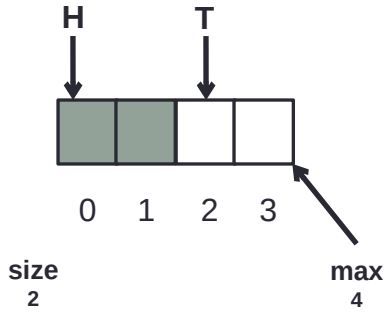


ADT Queue (Array): Implementation

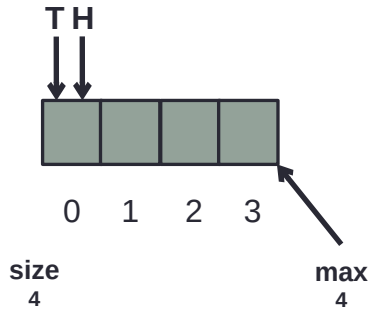
```
public boolean full () {
    return size == maxsize;
}

public int length () {
    return size;
}
```

ADT Queue (Array): Implementation

```
public boolean full () {
    return size == maxsize;
}

public int length () {
    return size;
}
```



false



true

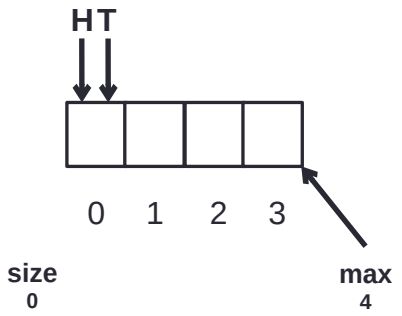
ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

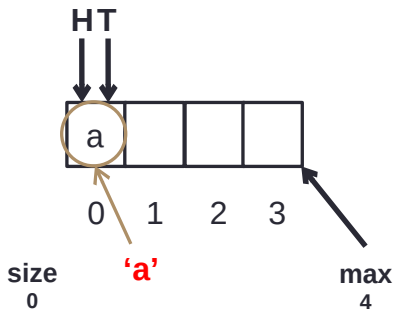
Example #1



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #1

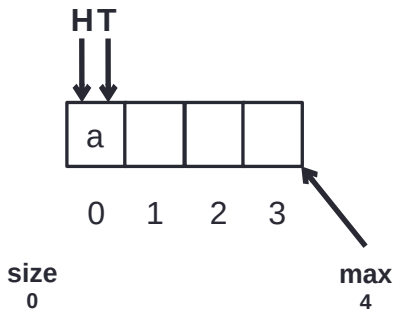


ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #1

(0 + 1) % 4 = 1 % 4 = 1

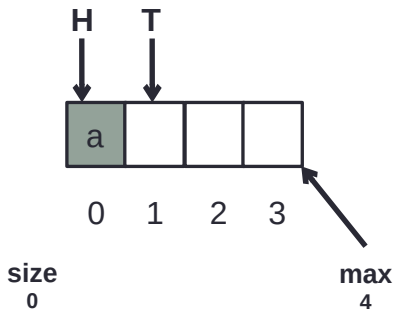


ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #1

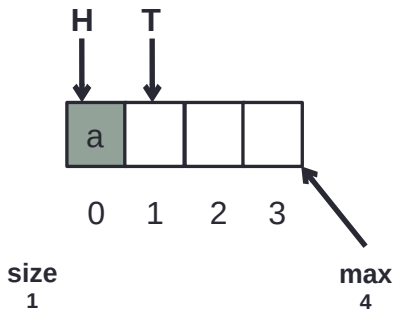
(0 + 1) % 4 = 1 % 4 = 1



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

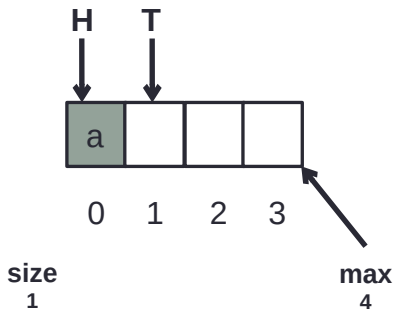
Example #1



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

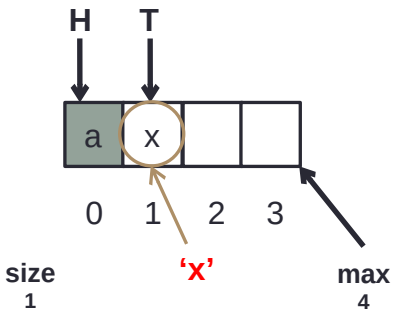
Example #2



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #2

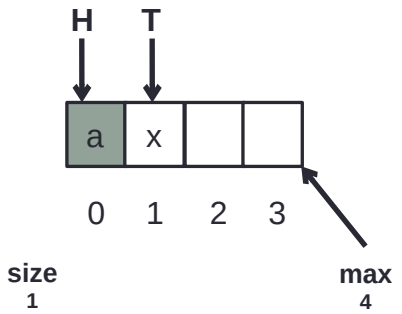


ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #2

(1 + 1) % 4 = 2 % 4 = 2

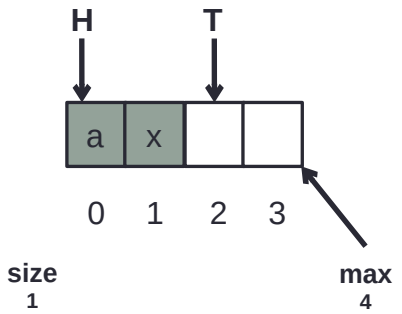


ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

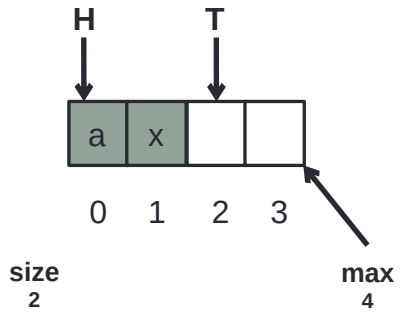
Example #2

(1 + 1) % 4 = 2 % 4 = 2



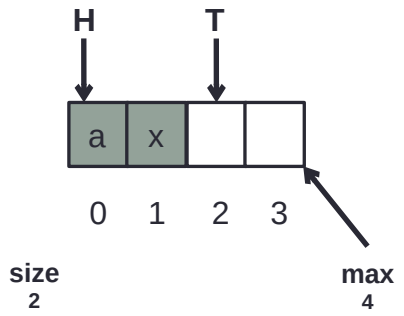
ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```



ADT Queue (Array): Implementation

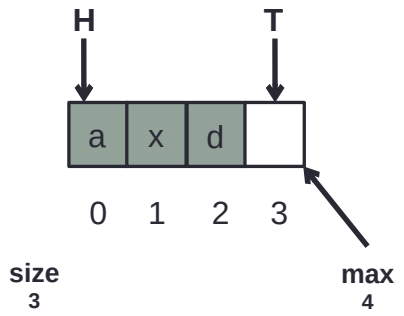
```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

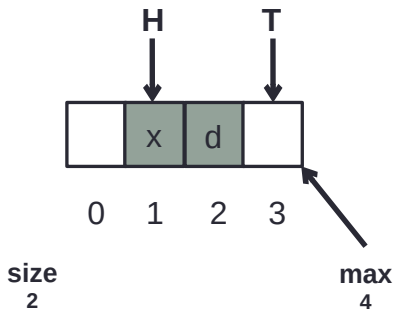
After one Enqueue



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

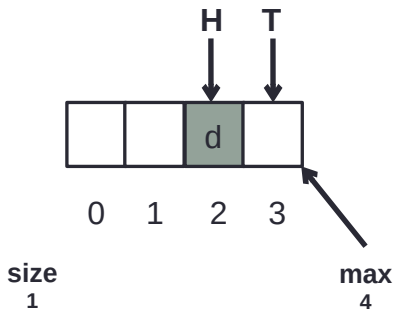
After one Serve



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

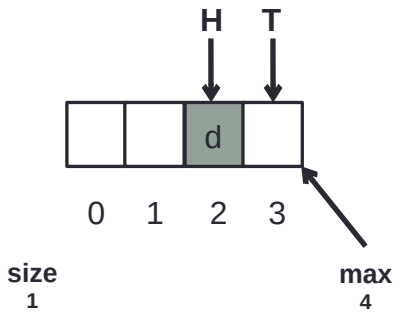
After another Serve



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

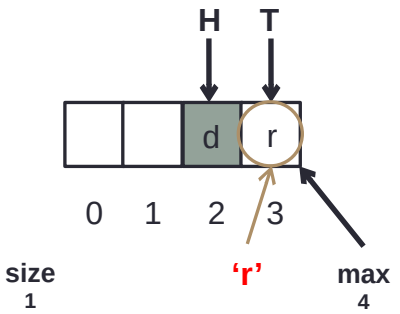
Example #3



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #3

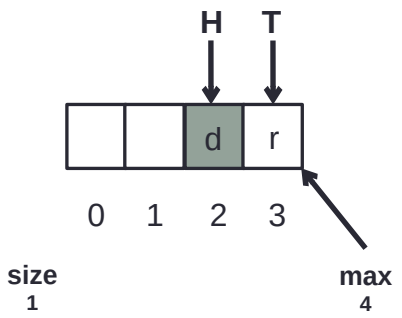


ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #3

(3 + 1) % 4 = 4 % 4 = 0

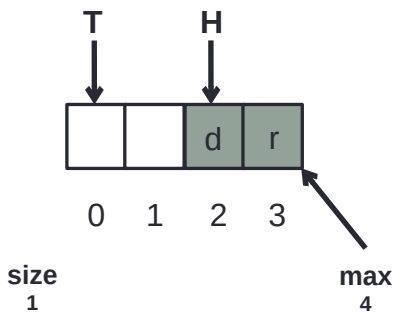


ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #3

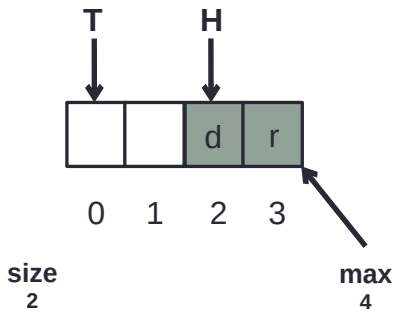
(3 + 1) % 4 = 4 % 4 = 0



ADT Queue (Array): Implementation

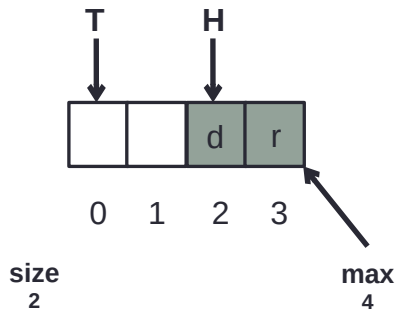
```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

Example #3



ADT Queue (Array): Implementation

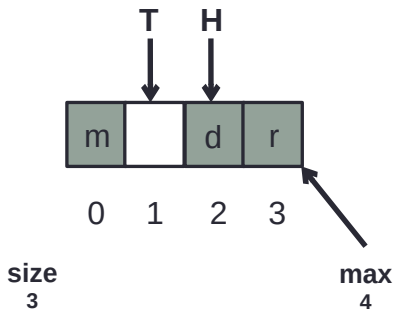
```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

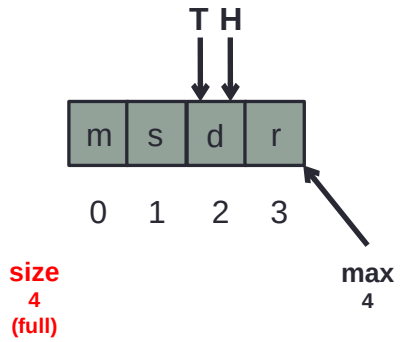
After one Enqueue



ADT Queue (Array): Implementation

```
public void enqueue(T e) {
    nodes[tail] = e;
    tail = (tail + 1) % maxsize;
    size++;
}
```

After another Enqueue



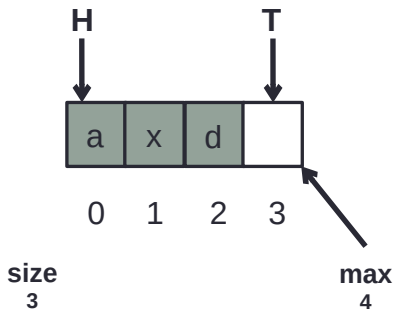
ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

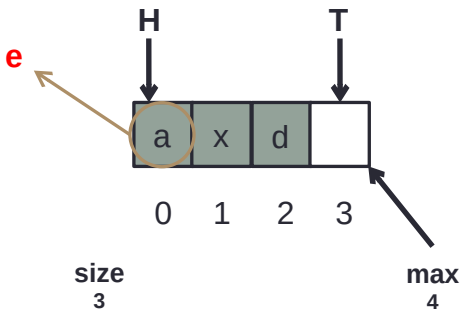
Example #1



ADT Queue (Array): Implementation

```
public T serve () {
    T e = nodes[head];
    head = (head + 1) % maxsize;
    size--;
    return e;
}

}
```

Example #1

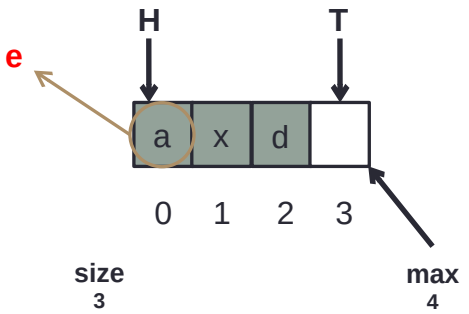


ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

Example #1

(0 + 1) % 4 = 1 % 4 = 1

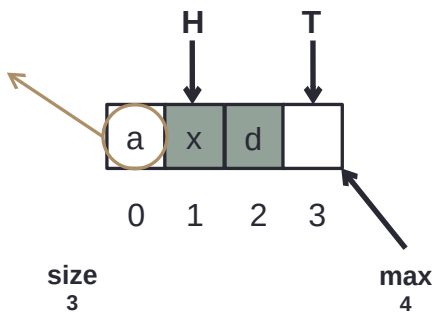


ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

Example #1

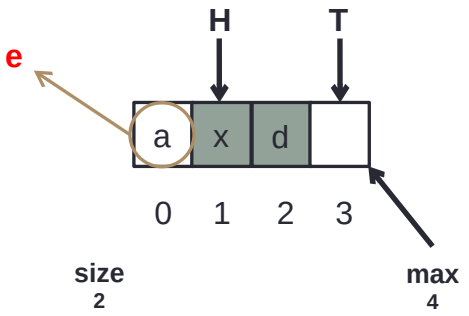
(0 + 1) % 4 = 1 % 4 = 1



ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

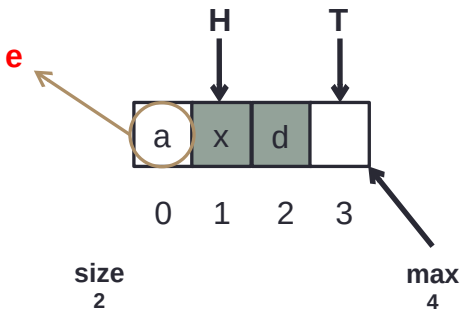
Example #1



ADT Queue (Array): Implementation

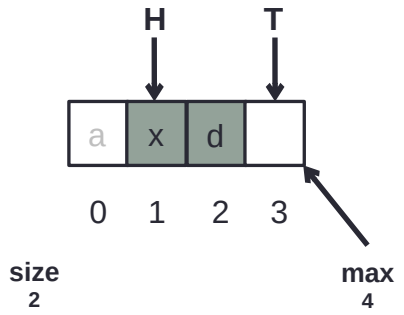
```
public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

Example #1



ADT Queue (Array): Implementation

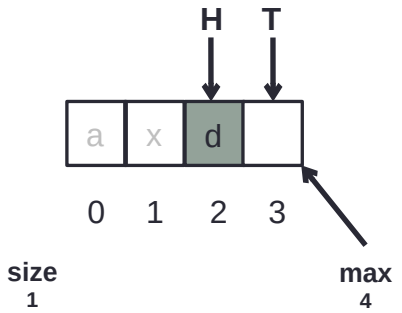
```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```



ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

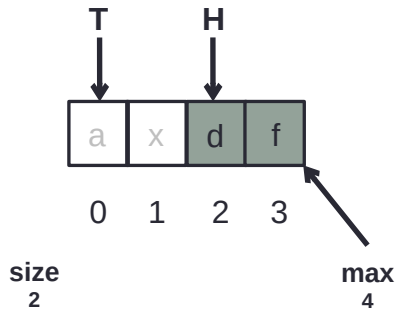
After another Serve



ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

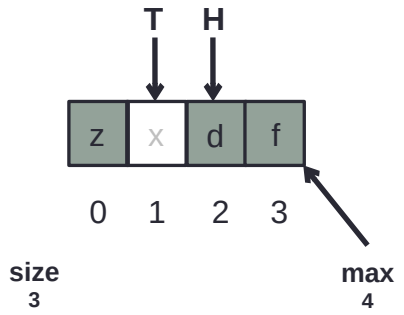
After one Enqueue



ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

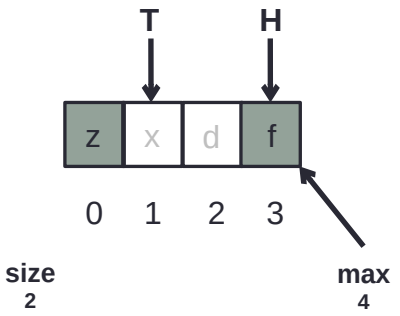
After another Enqueue



ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

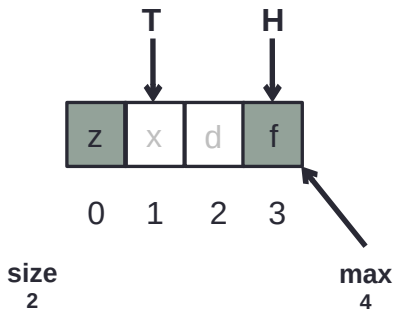
After one Serve



ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

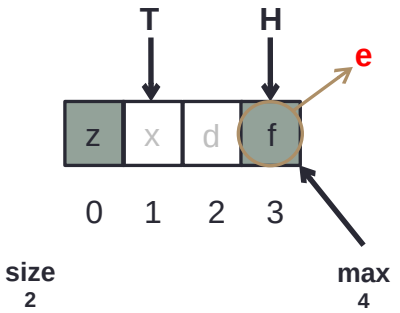
Example #2



ADT Queue (Array): Implementation

```
public T serve () {
    T e = nodes[head];
    head = (head + 1) % maxsize;
    size--;
    return e;
}

}
```

Example #2

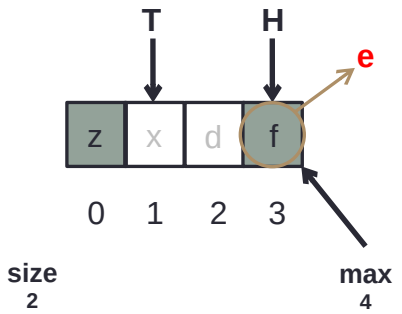


ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

Example #2

(3 + 1) % 4 = 4 % 4 = 0

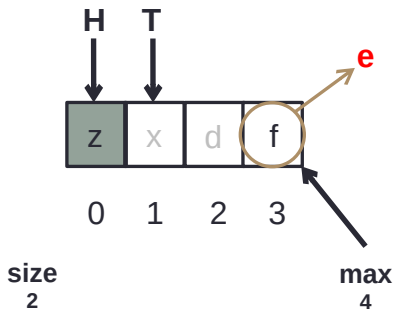


ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

Example #2

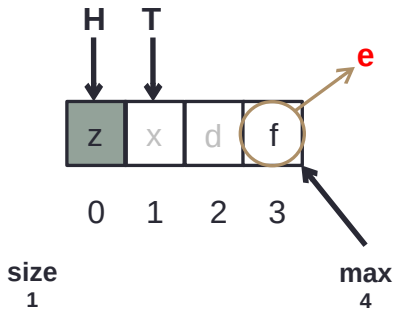
(3 + 1) % 4 = 4 % 4 = 0



ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

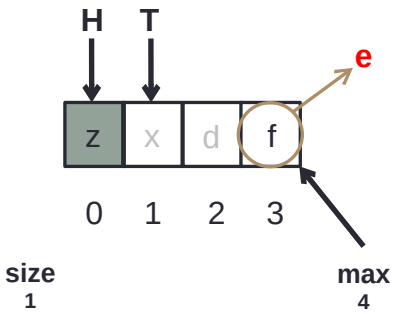
Example #2



ADT Queue (Array): Implementation

```
public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

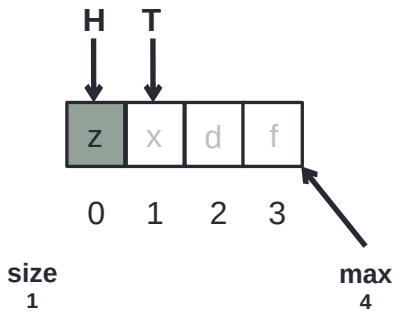
Example #2



ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

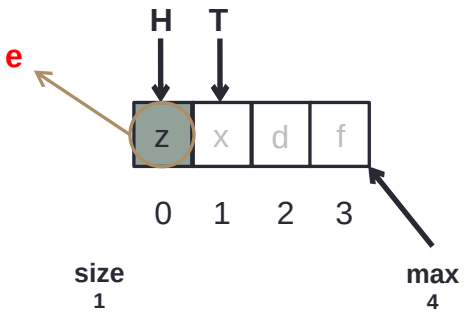
Example #3



ADT Queue (Array): Implementation

```
public T serve () {
    T e = nodes[head];
    head = (head + 1) % maxsize;
    size--;
    return e;
}

}
```

Example #3

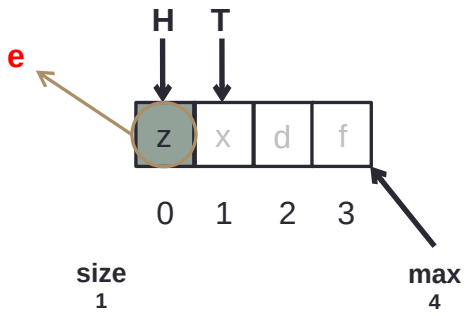


ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

Example #3

(0 + 1) % 4 = 1 % 4 = 1

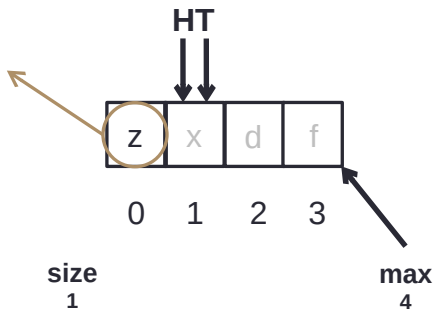


ADT Queue (Array): Implementation

```
public T serve () {
      T e = nodes[head];
      head = (head + 1) % maxsize;
      size--;
      return e;
  }

}
```

Example #3

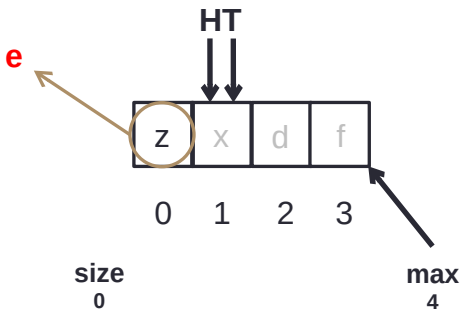
(0 + 1) % 4 = 1 % 4 = 1



ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

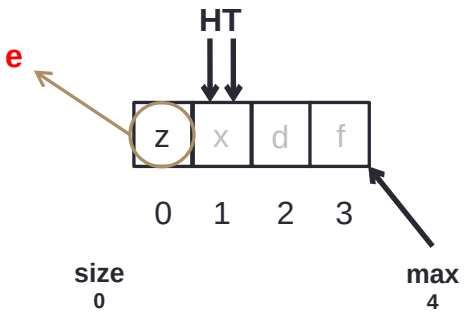
Example #3



ADT Queue (Array): Implementation

```
public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
}

}
```

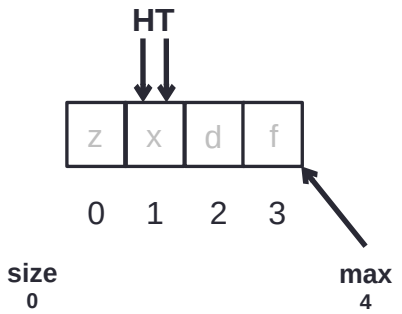
Example #3



ADT Queue (Array): Implementation

```
  public T serve () {
        T e = nodes[head];
        head = (head + 1) % maxsize;
        size--;
        return e;
  }

}
```

Example #3



Exercice

- Write the method enquiry that returns the data at the head of the queue without changing it.
  - As user of the ADT Queue<T>
  - As a member of the class LinkedQueue<T>
  - As a member of the class ArrayQueue<T>

Static Method Enquiry (LinkedQueue/ArrayQueue)

```
public static<T> T enquiry(Queue<T> q) {
      T data = q.serve();
      q.enqueue(data);
      for(int i = 0; i < q.length() – 1; i++)
          q.enqueue(q.serve());
      return data;
}
```

Member Method Enquiry (LinkedQueue)

```
public T enquiry() {
        return head.data;
}
```

Member Method Enquiry (ArrayQueue)

```
public T enquiry() {
        return nodes[head];
}
```

Priority Queue

- Each data element has a priority associated with it. Highest priority item is served first.
- Real World Priority Queues: hospital emergency rooms (most sick patients treated first), events in a computer system, etc.
- Priority Queue can be viewed as:
  - **View 1:** Priority queue as an ordered list.
  - **View 2:** Priority queue as a set.

ADT Priority Queue

Elements: The elements are of type PQNode. Each node has in it a data element of generic type <Type> and a priority of type Priority (which could be int type).

Structure: the elements are linearly arranged, and may be ordered according to a priority value, highest priority element is called the front or head.

Domain: the number of nodes in the queue is bounded therefore the domain is finite. Type of elements: PriorityQueue

ADT Priority Queue

Operations:

1. **Method** Enqueue (Type e, Priority p)
   requires: PQ is not full. **input:** e, p.
   results: Element e is added to the queue according to its priority. **output:** none.
2. **Method** Serve (PQElement<Type> pqe)
   requires: PQ is not empty. **input**: None
   results: the element and the priority at the head of PQ is removed and returned. **output**: pqe.
3. **Method** Length (int length)
   input: **results**: The number of element in the PQ is returned. **output**: length.

ADT Priority Queue

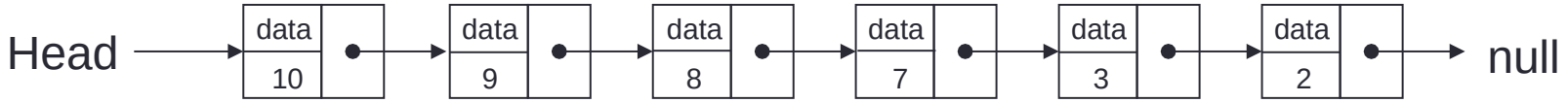
Operations:

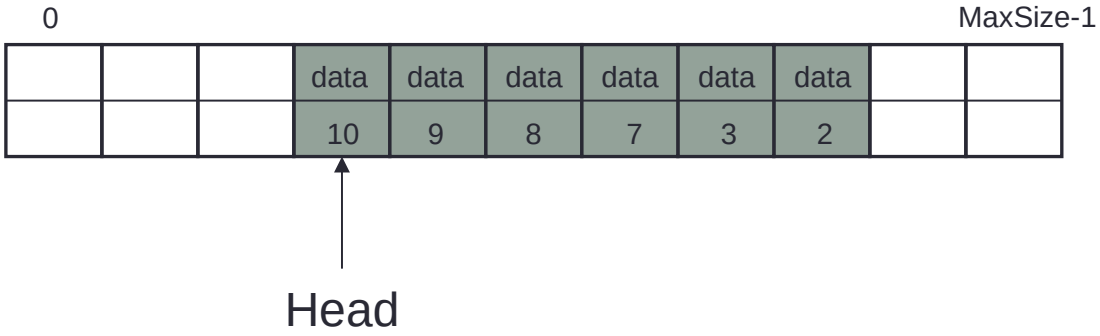
4. **Method** Full (boolean flag).

   requires: **input**:

   results: If PQ is full then flag is set to true, otherwise flag is set to false. **output**: flag.

ADT Priority Queue





ADT Priority Queue (Linked-List): Element

```
public class PQNode<T> {
  public T data;
  public Priority priority;
  public PQNode<T> next;

  public PQNode() {
        next = null;
  }

  public PQNode(T e, Priority p) {
        data = e;
        priority = p;
  }

  // Setters/Getters?
}
```

ADT Priority Queue (Linked-List): Element (int Priority)

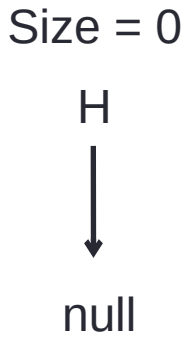
```
public class PQNode<T> {
  public T data;
  public int priority;
  public PQNode<T> next;

  public PQNode() {
        next = null;
  }

  public PQNode(T e, int p) {
        data = e;
        priority = p;
  }

  // Setters/Getters?
}
```

ADT Priority Queue (Linked-List): Representation

```
public class LinkedPQ<T> {
  private int size;
  private PQNode<T> head;


  /* tail is of no use here. */
  public LinkedPQ() {
        head = null;
        size = 0;
  }
```

ADT Priority Queue (Linked-List): Representation

```
public class LinkedPQ<T> {
  private int size;
  private PQNode<T> head;


  /* tail is of no use here. */
  public LinkedPQ() {
        head = null;
        size = 0;
  }
```



ADT Priority Queue (Linked-List): Implementation

```
public int length (){
    return size;
}
```

```
public boolean full () {
    return false;
}
```

ADT Priority Queue (Linked-List): Implementation

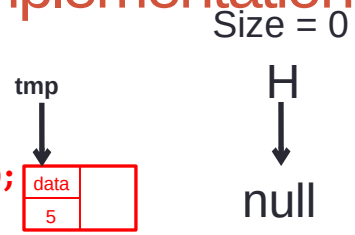
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

ADT Priority Queue (Linked-List): Implementation

```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

Example #1

ADT Priority Queue (Linked-List): Implementation

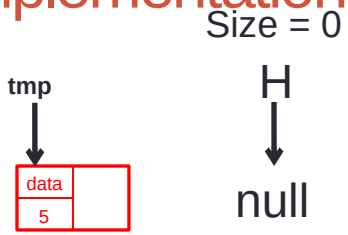



```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

Example #1

ADT Priority Queue (Linked-List): Implementation

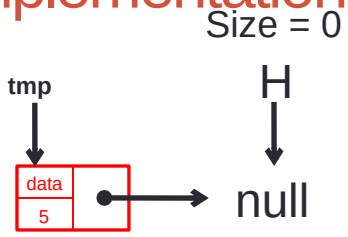
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #1

ADT Priority Queue (Linked-List): Implementation

```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

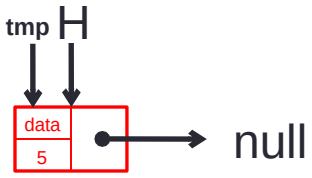


Example #1

ADT Priority Queue (Linked-List): Implementation

```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

Size = 0

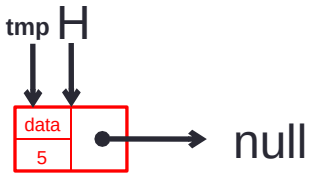


Example #1

ADT Priority Queue (Linked-List): Implementation

```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

Size = 1

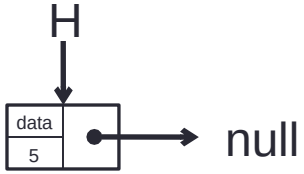


Example #1

ADT Priority Queue (Linked-List): Implementation

```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

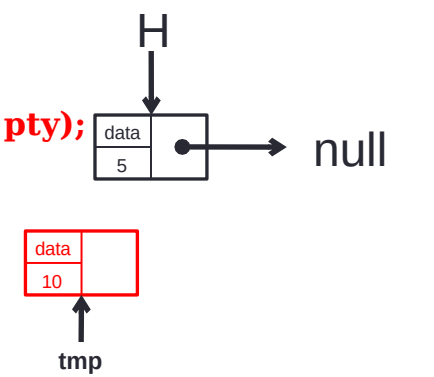
Size = 1



Example #2

ADT Priority Queue (Linked-List): Implementation

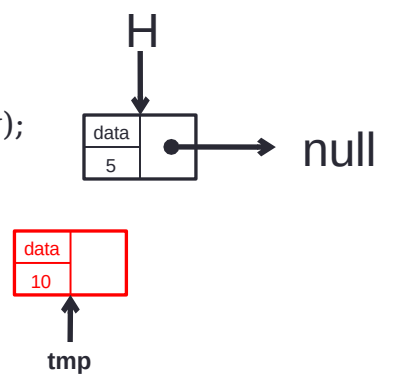
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #2

ADT Priority Queue (Linked-List): Implementation

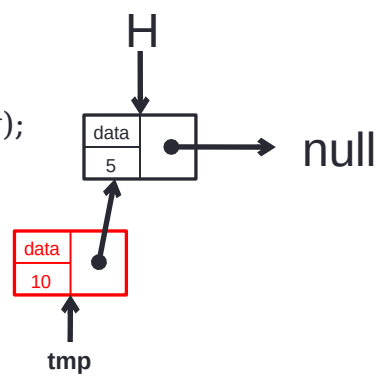
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #2

ADT Priority Queue (Linked-List): Implementation

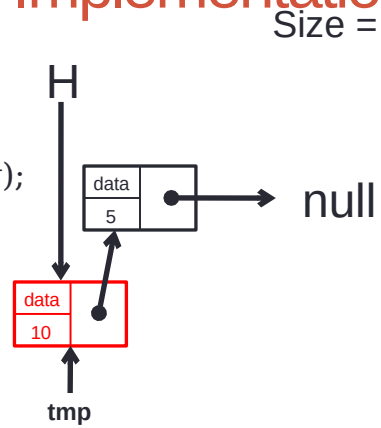
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #2

ADT Priority Queue (Linked-List): Implementation

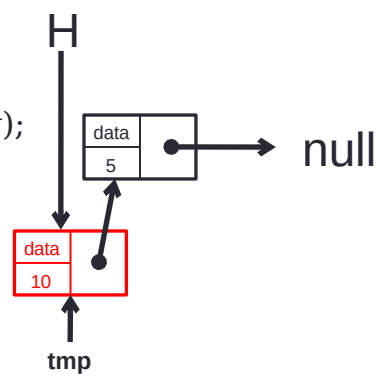
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #2

ADT Priority Queue (Linked-List): Implementation

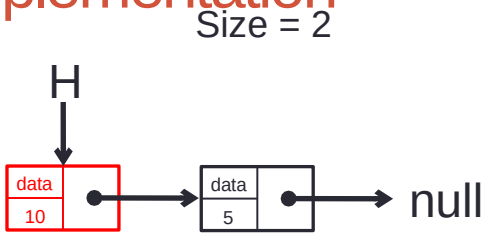
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #2

ADT Priority Queue (Linked-List): Implementation

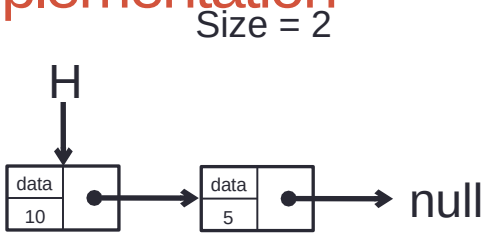
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #2

ADT Priority Queue (Linked-List): Implementation

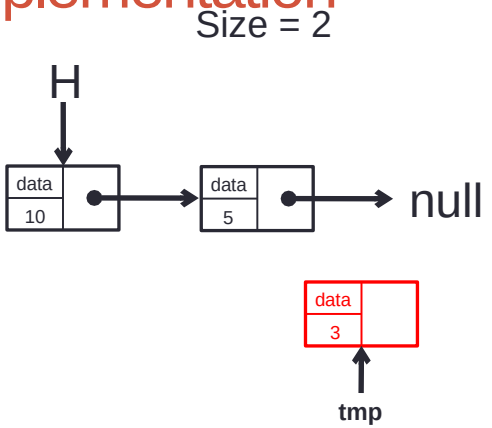
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

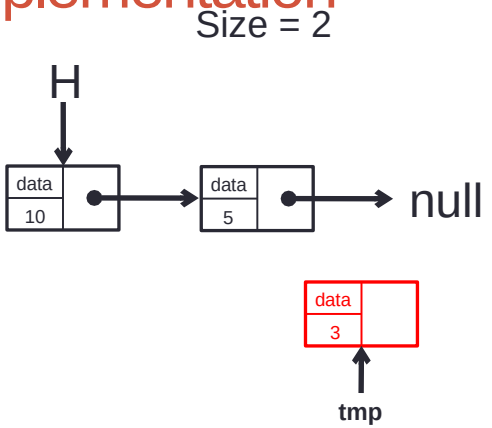
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

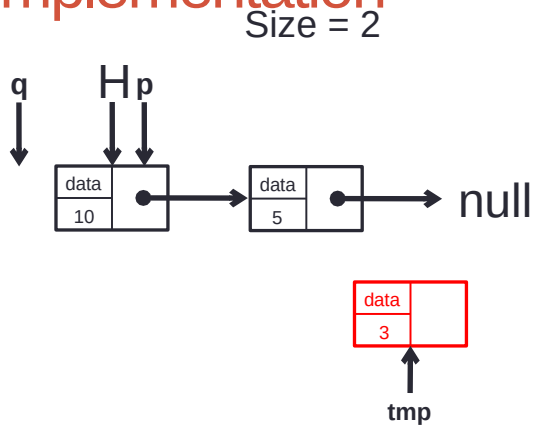
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

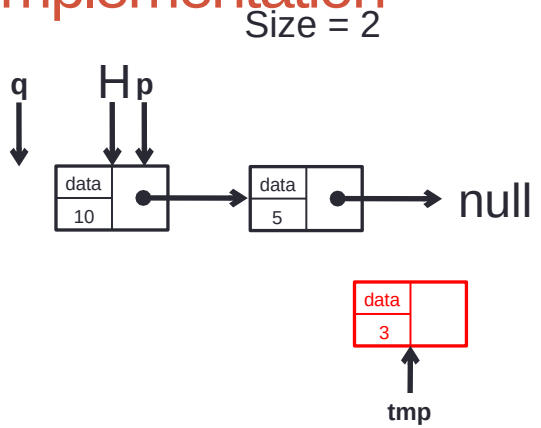
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

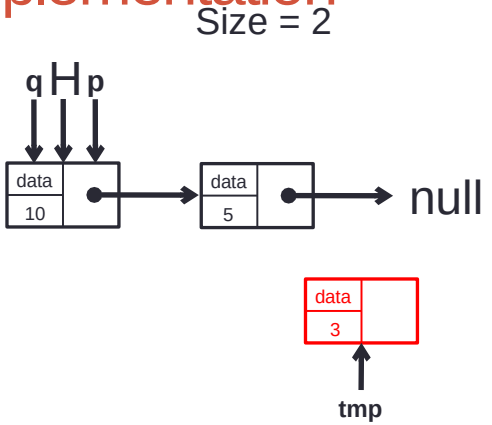
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

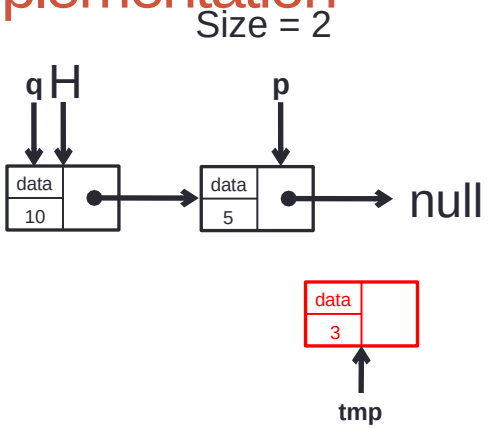
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

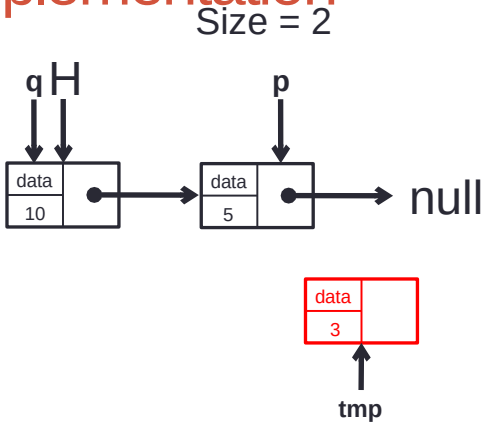
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

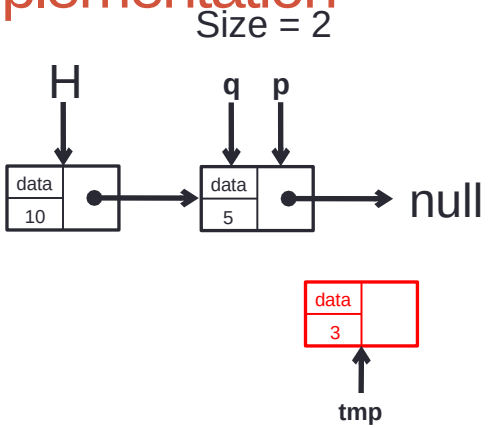
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

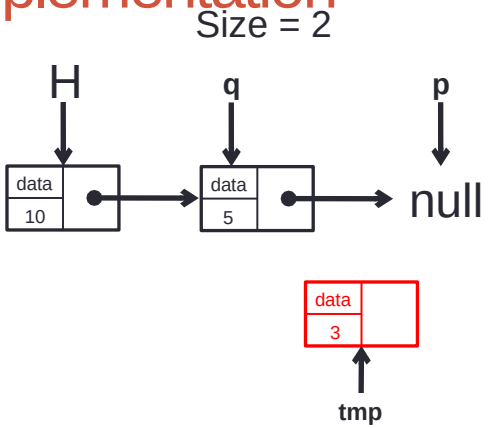
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

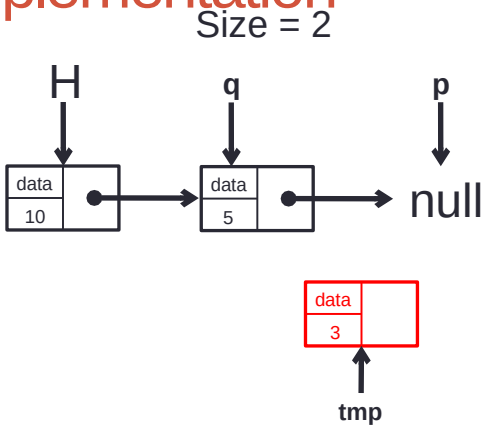
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

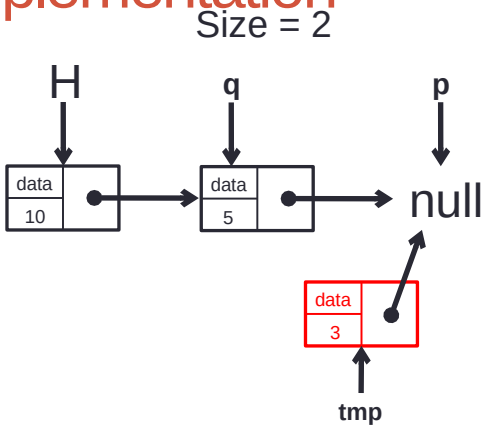
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

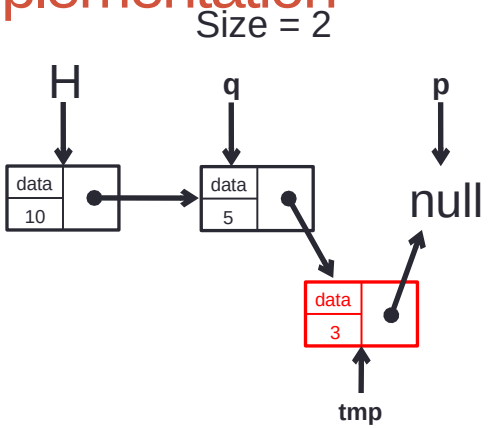
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

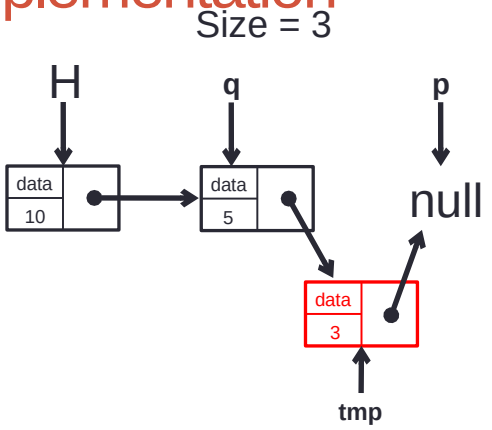
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

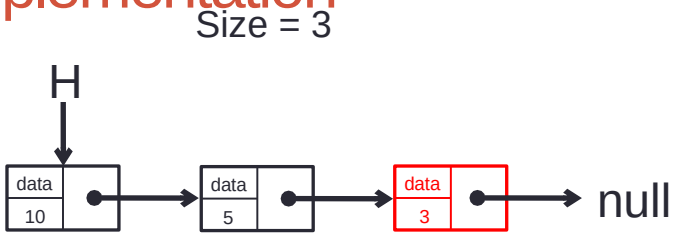
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

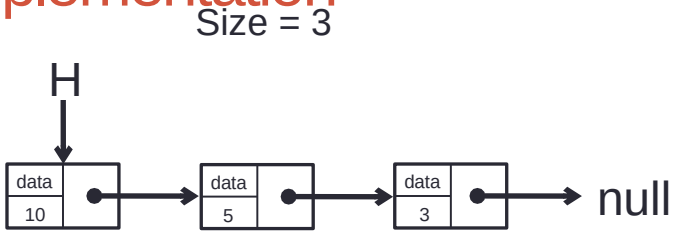
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #3

ADT Priority Queue (Linked-List): Implementation

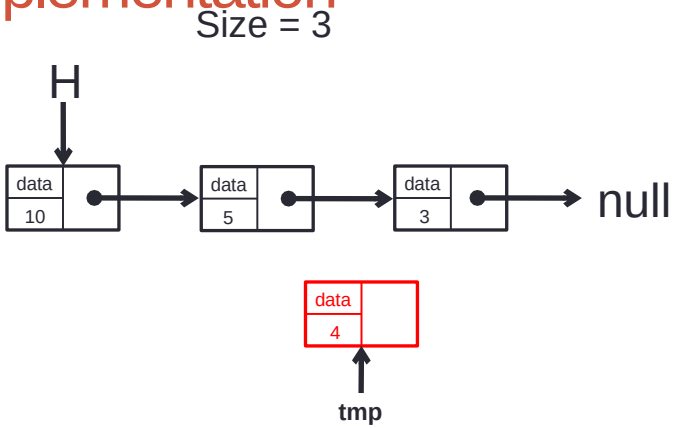
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

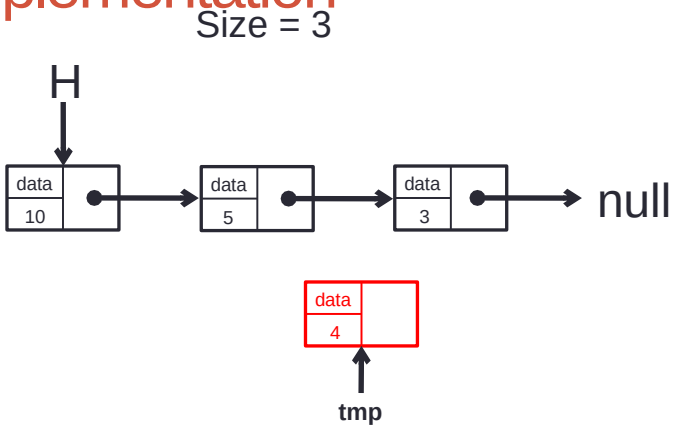
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

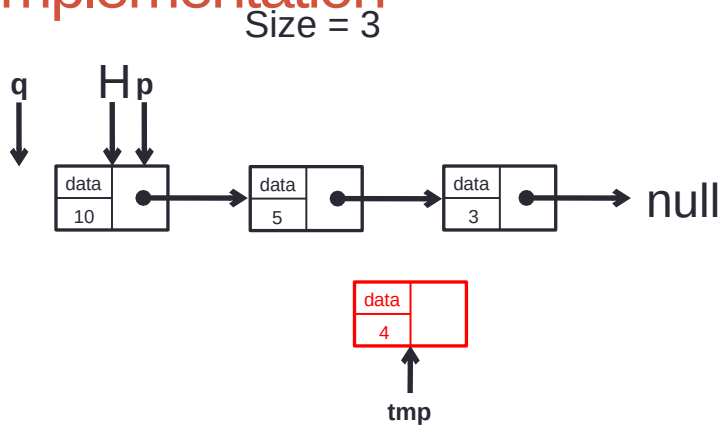
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

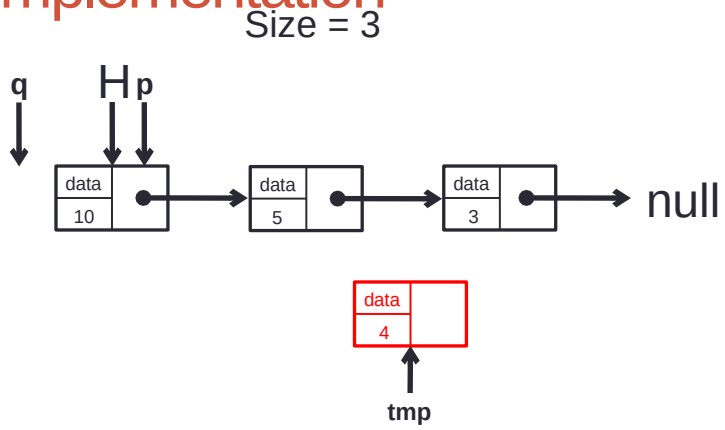
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

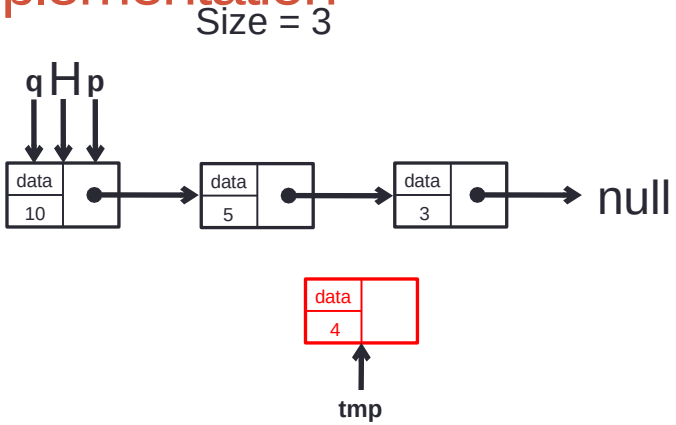
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

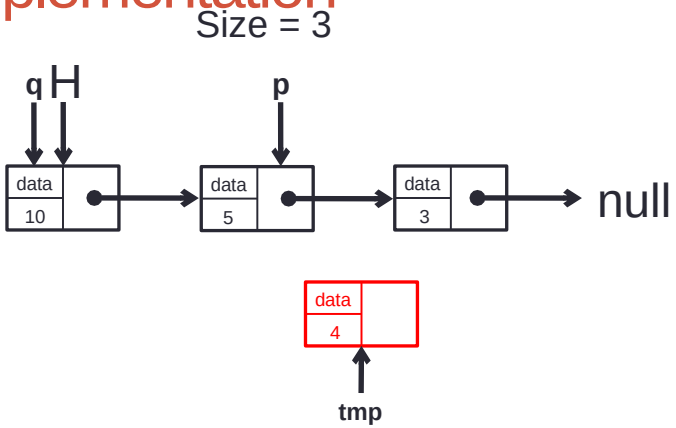
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

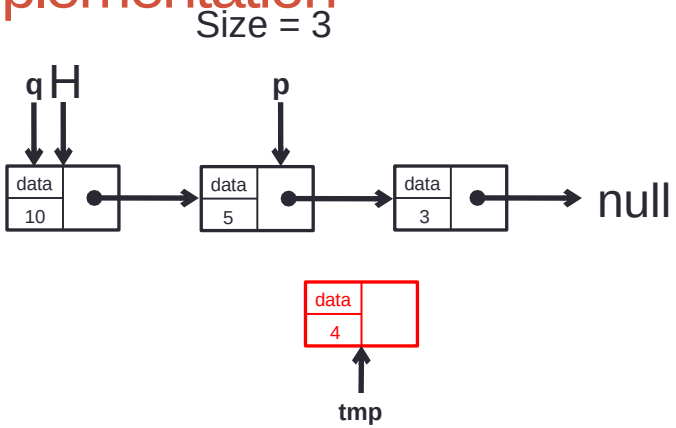
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

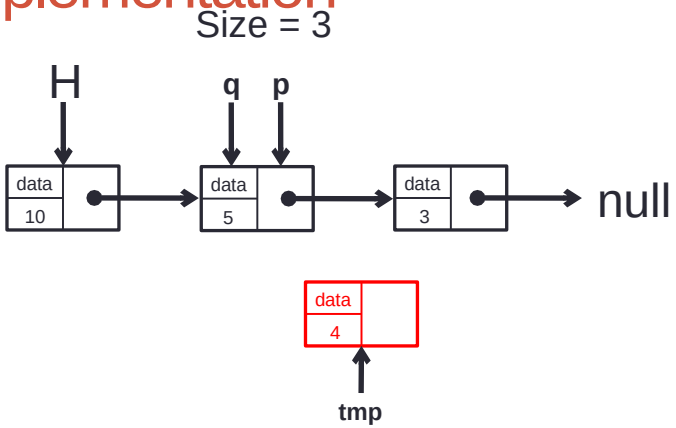
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

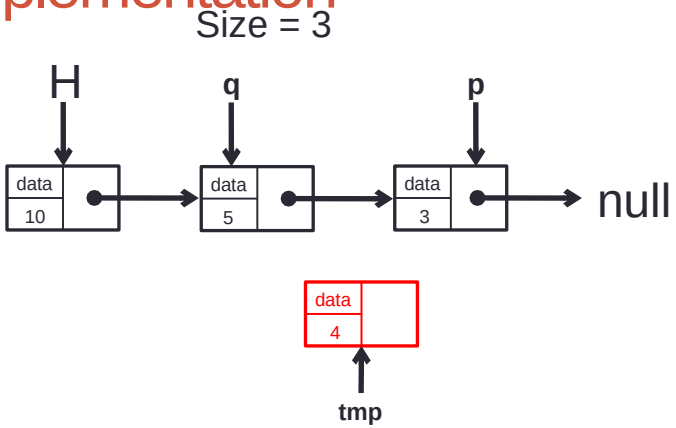
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

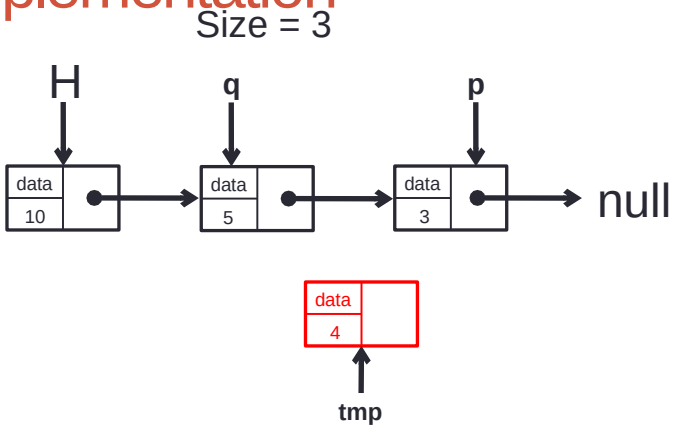
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

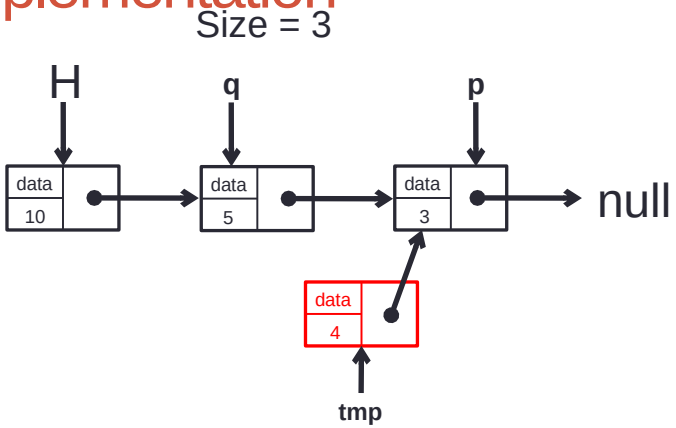
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

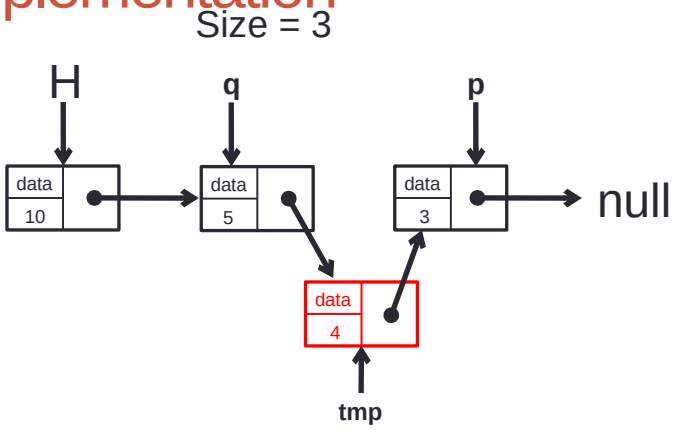
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

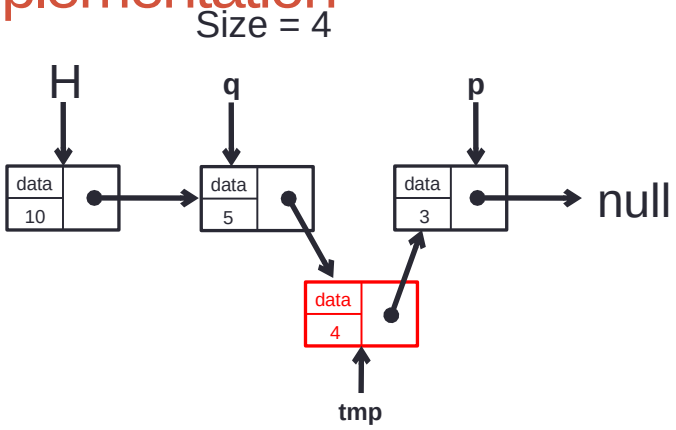
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

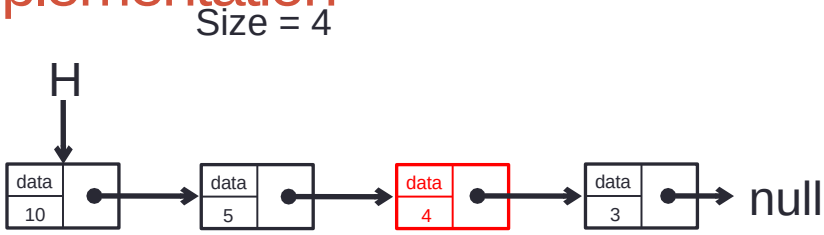
```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```



Example #4

ADT Priority Queue (Linked-List): Implementation

```
public void enqueue(T e, int pty) {
        PQNode<T> tmp = new PQNode<T>(e, pty);
        if((size == 0) || (pty > head.priority)) {
                tmp.next = head;
                head = tmp;
        }
        else {
                PQNode<T> p = head;
                PQNode<T> q = null;
                while((p != null) && (pty <= p.priority)) {
                        q = p;
                        p = p.next;
                }
                tmp.next = p;
                q.next = tmp;
        }
        size++;
}
```

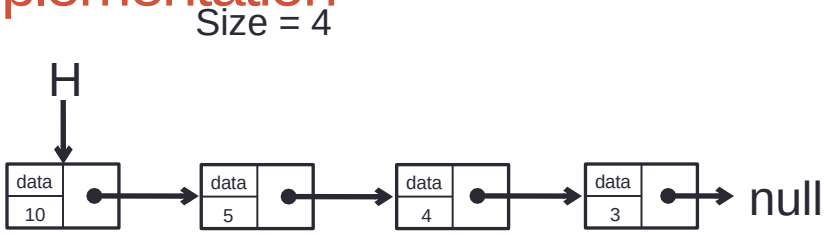


Example #4

ADT Priority Queue (Linked-List): Implementation

```
  public PQElement<T> serve(){
  PQNode<T> node = head;
  PQElement<T> pqe=new PQElement<T>(node.data,node.p);
  head = head.next;
  size--;
  return pqe;
  }

 public class PQElement<T>
 {
 public T data;
 public Priority p;
 public PQElement(T e, Priority pr){
  data=e;
  p=pr;
}
```

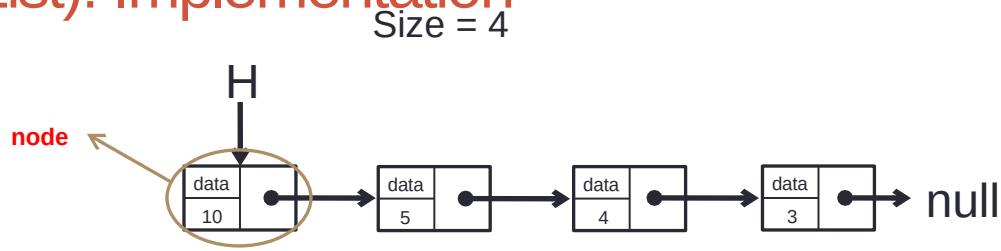
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #1

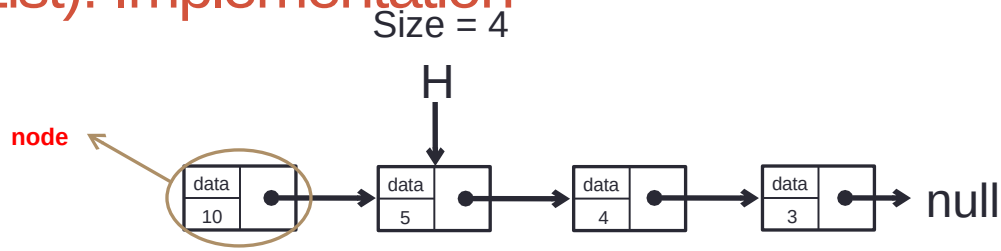
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #1

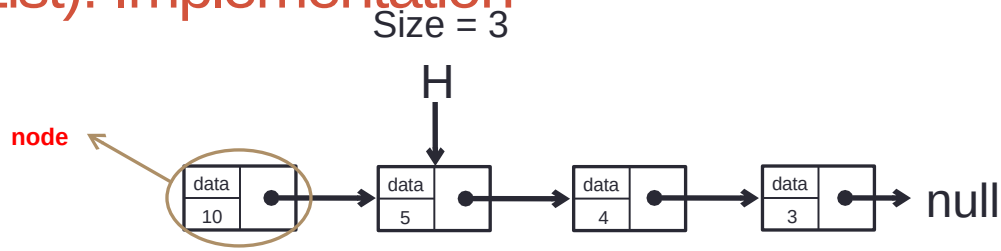
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #1

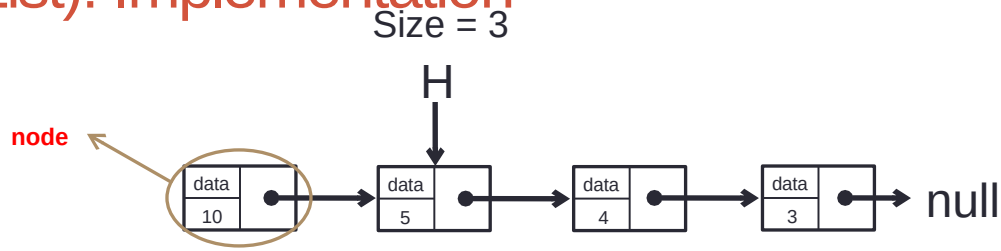
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #1

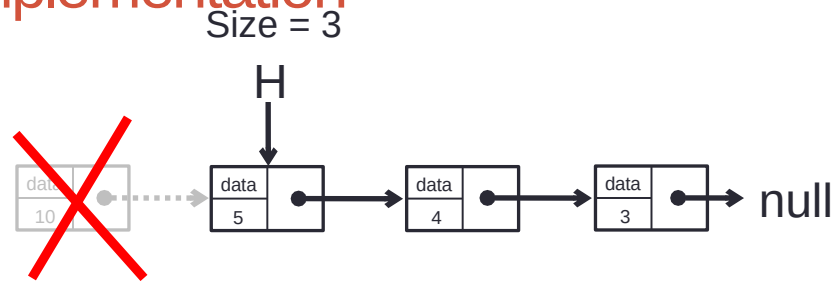
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #1

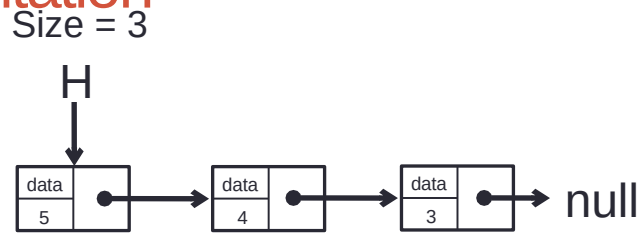
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #1

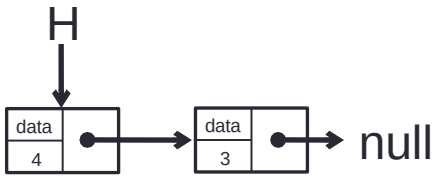
ADT Priority Queue (Linked-List): Implementation




```
public PQElement<T> serve(){
PQNode<T> node = head;
PQElement<T> pqe=new PQElement<T>(node.data,node.p);
head = head.next;
size--;
return pqe;
}
```

ADT Priority Queue (Linked-List): Implementation

Size = 2

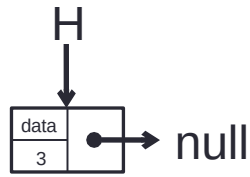



```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Another serve

ADT Priority Queue (Linked-List): Implementation

Size = 1

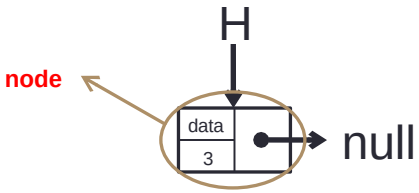



```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Yet another serve

ADT Priority Queue (Linked-List): Implementation

Size = 1

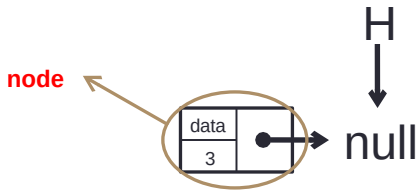



```
public PQElement<T> serve(){
        PQNode<T> node = head;
        PQElement<T> pqe=new
   PQElement<T>(node.data,node.p);
        head = head.next;
        size--;
        return pqe;
   }
```

Example #3

ADT Priority Queue (Linked-List): Implementation

Size = 1

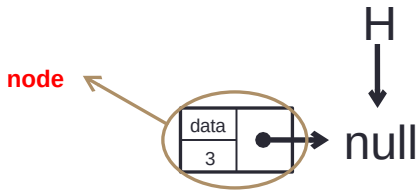


Example #3

```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

ADT Priority Queue (Linked-List): Implementation

Size = 0

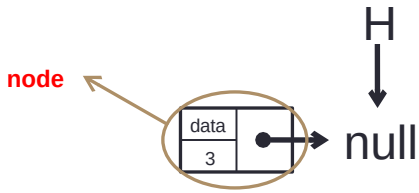



```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #3

ADT Priority Queue (Linked-List): Implementation

Size = 0




```
public PQElement<T> serve(){
    PQNode<T> node = head;
    PQElement<T> pqe=new PQElement<T>(node.data,node.p);
    head = head.next;
    size--;
    return pqe;
}
```

Example #3

ADT Priority Queue (Linked-List): Implementation

Size = 0

Example #3

```
public PQElement<T> serve(){
        PQNode<T> node = head;
        PQElement<T> pqe=new PQElement<T>(node.data,node.p);
        head = head.next;
        size--;
        return pqe;
  }
```

ADT Priority Queue

- Implementations
  - Linked List: Enqueue is O(n), Serve is O(1).
  - Array Implementation: Enqueue is O(1), Serve is O(1).
  - Heap: Enqueue is O(log n), Serve is O(log n) Heaps to be discussed later.

Double-Ended Queues

- Double ended queue (or a **deque**) supports insertion and deletion at both the front and the tail of the queue.
- The first element is called head and the last element is called tail.
- Supports operations: addFirst( ), addLast(), removeFirst( ) and removeLast( ).
- Can be used in place of a queue or a stack.

Double-Ended Queues

Operations: (Assume all operations are performed on deque DQ)

1. **Method** addFirst (Type e)
   requires: DQ is not full. **input:** e.
   results: Element e is added to DQ as first element. **output:** none.
2. **Method** addLast (Type e)
   requires: DQ is not full. **input**: e
   results: Element e is added to DQ as last element. **output**: none.
3. **Method** removeFirst (Type e)
   requires: DQ is not empty. **input**: none **results**: Removes and returns the first element of DQ. **output**: e.

Double-Ended Queues

4. **Method** removeLast (Type e)
   requires: DQ is not empty. **input:** none.
   results: Removes and returns the last element of DQ. **output:** e.
5. **Method** getFirst (Type e)
   requires: DQ is not empty. **input**: none
   results: Returns the first element of DQ. **output**: e.
6. **Method** getLast (Type e)
   requires: DQ is not empty. **input**: none
   results: Returns the last element of DQ. **output**: e
7. **Method** size (int x)
   input: none **results**: Returns the number of elements in DQ.
   output: x

Double-Ended Queues

8. **Method** empty (boolean x)

   input: none **results**: if DQ is empty returns x as true otherwise false. **output**: x

Complexity so far?

Operation	Queue (LL)	Queue (CA)	Priority Queue (LL)	Priority Queue (CA)
Full	O(1)	O(1)	O(1)	O(1)
Length	O(1)	O(1)	O(1)	O(1)
Enqueue	O(1)	O(1)	O(n)	O(n)
Serve	O(1)	O(1)	O(1)	O(1)

Complexity so far?

Operation	Double-Ended Queue (LL)	Double-Ended Queue (CA)	Double-Ended Queue (DLL)
AddFirst			
AddLast			
RemoveFirst			
RemoveLast	O(n)		O(1)
GetFirst			
GetLast			
Size			
Empty			

ToDo

- Read 5.2, 5.3 of the Textbook.
- Add “int length()” method in the LinkedQueue class with O(n) complexity.
- Add “int length(ArrayQueue<T> q) ” in the Test class of ArrayQueue. The Queue must remain unchanged after the operation.
- Add “T enquiry(ArrayQueue<T> q) ” in the Test class of ArrayQueue. It should return the data of the head without changing the queue at the end of the call.
- Implement DQueue (Double-ended queue) using a Java class using Linked-List.
- Test this DQueue using a test Class.