Data Structure and Algorithm Practicals

6. Demonstration of Circular Queue

```
var Queue = function(maxSize){
	this.queue = [];

	this.reset = function(){
		this.tail = -1;
		this.head = -1;
	};

	this.reset();
	this.maxSize = maxSize || Queue.MAX_SIZE;

	this.increment = function(number){
		return (number + 1) % this.maxSize;
	};
};

Queue.MAX_SIZE = Math.pow(2, 53) - 1;

Queue.prototype.enQueue = function(record){

	if(this.isFull()){
		throw new Error("Queue is full can't add new records");
	}

	if(this.isEmpty()){
		this.head = this.increment(this.head);
	}

	this.tail = this.increment(this.tail);
	//console.log("tail", this.tail);
	this.queue[this.tail] = record;


};

Queue.prototype.setMaxSize = function(maxSize){
	this.maxSize = maxSize;
};

Queue.prototype.push = Queue.prototype.enQueue;
Queue.prototype.insert = Queue.prototype.enQueue;

Queue.prototype.isFull = function(){
	return this.increment(this.tail) === this.head;
};
```

```
Queue.prototype.deQueue = function(){
	if(this.isEmpty()){
		throw new Error("Can't remove element from an empty Queue");
	}

	// removing from the begining of the head
	var removedRecord = this.queue[this.head];
	this.queue[this.head] = null;

	if(this.tail === this.head){
		this.reset();
	}else{
	  // if there are more records increase head.
		this.head = this.increment( this.head );
	}

	return removedRecord;
};

Queue.prototype.pop = Queue.prototype.deQueue;

Queue.prototype.front = function(){

	return this.queue[this.head] || null;
};

Queue.prototype.peak = Queue.prototype.front;

Queue.prototype.isEmpty = function(){
	return this.tail === -1 && this.head === -1;
};

Queue.prototype.print = function(){
	for(var i= this.head; i <= this.tail; i++){
		console.log(this.queue[i]);
	}
};

var q = new Queue(5);
q.enQueue(1);
q.enQueue(2);
q.enQueue(3);
q.enQueue(4);
 q.deQueue();
 q.deQueue();
 q.deQueue();
```

```
q.enQueue(5);
q.enQueue(6);
q.enQueue(7);
q.enQueue(8);
 q.deQueue();
 q.deQueue();
 q.deQueue();
 q.deQueue();


//q.print();

//var el = q.deQueue();
//console.log("removed element" + el);
//console.clear();
q.print();

console.log("head", q.head);
console.log("tail", q.tail);
console.log(q.queue);
```