Chapitre 2 Type générique et enum

En TypeScript, vous avez la possibilité de définir une collection de constantes à l'aide du mot réservé enum :

```
enum Status{
    Published,
    Unpublished,
    Draft
}

// Une variable peut avoir un type enum
let state : Status = state.Published ;
```

Typage générique

Il permet de définir pour les fonctions, classes ou interfaces un type non pré-défini que l'on précisera a posteriori. C'est lors de l'appel de la fonction, l'interface ou la classe que le type sera précisé :

```
// T type non pré-défini
function fusion<T>(a: T[], b : T[]): T[]{
    return a.contact(b);
}
// On précise le type T à l'appelle de la fonction
let c = funsion<string>(['a','b','c'], ['d','e', 'f']);
```

Exercice 1

Implémentez une Queue en TypeScript en définissant une classe Queue et en utilisant le typage générique. Vous devez dans cette classe redéfinir les méthodes push et pop :

```
class Queue<T> {
}

let queue = new Queue<number>();
queue.push(1);
queue.push(2);
queue.push(3);
queue.push(4);
console.log(queue.pop()); // affiche 1
```