Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Algorithmique et structures de données

S11 Files d'attente

Auteurs :
Marc Roten

Professeur :
Rudolph Scheurer

4 mai 2018

Table des matières

1 IntQueueArray **2**

2 IntQueueChained **3**

3 Methode QueueChained<E> **4**

3.1 partie a Code de la classe 4

3.2 Code partie b 5

3.3 Code 6

Table des figures

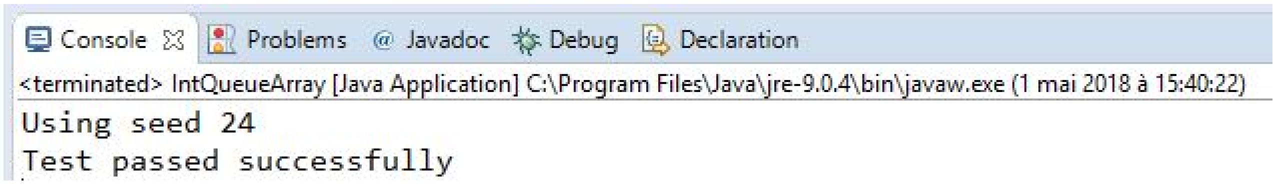
1 Résultat du mini test 2

2 Résultat du mini test 3

3 Résultat des tests de généricité 6

1 IntQueueArray

```
// PRE: !isEmpty()
public int      dequeue    ()          {
        int myTemp = buffer[front];
        front++;
        if(front==buffer.length) {
                /*
                 * circular logic
                 */
                front=0;
        }
        size--;
        return myTemp;
}
// ------------------------------------------
private void checkSize(){
        if (size<buffer.length)return;
        int[] CopyBuffer =new int[2*buffer.length];
        int x;
        int FrontTempo =1;
        for (x =front;x <buffer.length;x++){
                CopyBuffer[FrontTempo]=buffer[x];
                FrontTempo++;
        }
        for (int j =0;j <front;j++){
        CopyBuffer[FrontTempo]=buffer[j];
        FrontTempo++;
        }
        buffer =CopyBuffer;
        front =1;
        back =--FrontTempo;
}
```

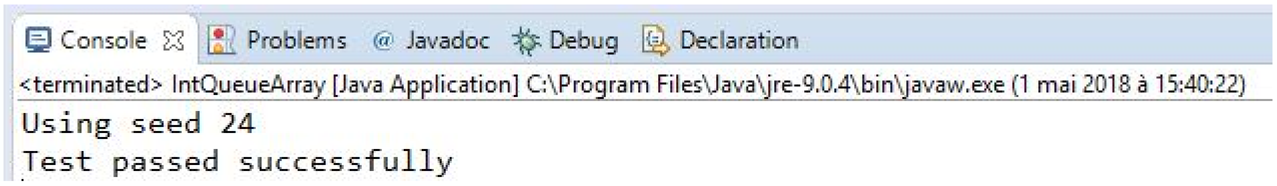


```
<terminated> IntQueueArray [Java Application] C:\Program Files\Java\jre-9.0.4\bin\javaw.exe (1 mai 2018 à 15:40:22)
Using seed 24
Test passed successfully
```

FIGURE 1 – Résultat du mini test

2 IntQueueChained

```
public void enqueue (int elt) {
        if(front==null && back == null) {
                front = new QueueNode(elt);
                back = front;
        }else {
                QueueNode QueueNodeInternal = new QueueNode(elt);
                back.next = QueueNodeInternal;
                back = QueueNodeInternal;
        }
}
```



```
<terminated> IntQueueArray [Java Application] C:\Program Files\Java\jre-9.0.4\bin\javaw.exe (1 mai 2018 à 15:40:22)
Using seed 24
Test passed successfully
```

FIGURE 2 – Résultat du mini test

3 Methode QueueChained<E>

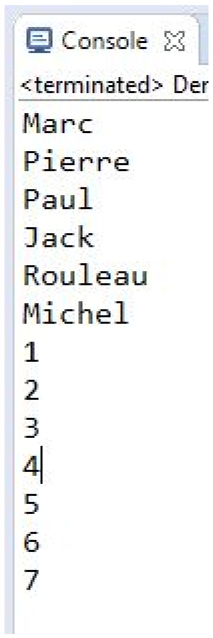
3.1 partie a Code de la classe

```
public class QueueChained<E> {

public          QueueChained() {};
        private IteratorQueue<E> Top;
        private IteratorQueue<E> Bot;
public void     enqueue (E elt) {
        if(Top==null && Bot == null) {
                Top = new IteratorQueue<E>(elt);
                Bot = Top;
        }else {
        IteratorQueue<E> NewItQueue= new IteratorQueue<E>(elt);
        Bot.prev = NewItQueue;
        Bot = NewItQueue;
        }
    };
        public boolean isEmpty () {
                return Bot ==null;
        };
        public E        consult () {
                return Top.elt;
        };
        public E        dequeue () {
                E res = Top.elt;
                if(Top==Bot) {
                        Bot = null;
                        Top = null;
                }else {
                        Top = Top.prev;
                }
                return res;
        };
        static class IteratorQueue<E>{
                E elt;
                IteratorQueue prev = null;
                IteratorQueue(E elt){
                        this.elt = elt;
                }
        }
}
```

3.2 Code partie b

```
public class Demo {
        static void demo(int n) {
                QueueChained<Integer> f;
                int i, sum=0;
                f = new QueueChained<Integer>();
                for (i=0; i<n; i++)
                        f.enqueue(i);
                while(! f.isEmpty())
                        sum = sum + f.dequeue();
                System.out.println(sum);
        }
        public static void main(String[] args) {
                QueueChained<String> f = new QueueChained<String>();
                f.enqueue("Marc");
                f.enqueue("Pierre");
                f.enqueue("Paul");
                f.enqueue("Jack");
                f.enqueue("Rouleau");
                f.enqueue("Michel");
                while(!f.isEmpty()) {
                        System.out.println(f.dequeue());
                }
                QueueChained<Integer> g = new QueueChained<Integer>();
                g.enqueue(1);
                g.enqueue(2);
                g.enqueue(3);
                g.enqueue(4);
                g.enqueue(5);
                g.enqueue(6);
                g.enqueue(7);
                while(!g.isEmpty()) {
                        System.out.println(g.dequeue());
                }
        }
}
```



```
Marc
Pierre
Paul
Jack
Rouleau
Michel
1
2
3
4
5
6
7
```

FIGURE 3 – Résultat des tests de généricité

3.3 Code

```
public class Demo {
        static void demo(int n) {
                ObjQueue f;
                int i, sum=0;
                f = new ObjQueue();
                for (i=0; i<n; i++)
                        f.enqueue(i);
                while(! f.isEmpty())
                        sum = sum + f.dequeue();
                System.out.println(sum);
        }
        public static void main(String[] args) {
                Demo.demo(20);
        }
}
```