

Proves unitàries amb JUnit

```
1 package junit.cua;
2
3 public class Cua {
4     final int MIDA_MAX;
5     int primer;
6     int darrer;
7     int mida;
8     int elements[];
9
10    public Cua(int midaMax) {
11        assert midaMax > 0;
12        MIDA_MAX = midaMax;
13        primer = 0;
14        darrer = 0;
15        mida = 0;
16        elements = new int[MIDA_MAX];
17    }
18
19    public boolean esBuida() {
20        checkRep();
21        return mida == 0;
22    }
23
24    public boolean esPlena() {
25        checkRep();
26        return mida == MIDA_MAX;
27    }
28
29    public void encuar(int element) throws Exception {
30        checkRep();
31        if (esPlena()) {
32            throw new Exception("La cua és plena");
33        }
34        elements[darrer] = element;
35        mida++;
36        darrer++;
37        if (darrer == MIDA_MAX) {
38            darrer = 0;
39        }
40    }
```

```
41
42    public int desencuar() throws Exception {
43        checkRep();
44        if (esBuida()) {
45            throw new Exception("La cua és buida");
46        }
47        int element = elements[primer];
48        mida--;
49        primer++;
50        if (primer == MIDA_MAX) {
51            primer = 0;
52        }
53        return element;
54    }
55
56    private void checkRep() {
57        assert 0 <= mida && mida <= MIDA_MAX;
58        if (primer < darrer) {
59            assert mida == darrer - primer;
60        } else if (primer > darrer) {
61            assert mida == MIDA_MAX - (primer - darrer);
62        } else {
63            assert mida == 0 || mida == MIDA_MAX;
64        }
65    }
66 }
67
```

Implementació de cua	Temps (en mil·lisegons) en encuar 100.000 elements i després desencuar-los
ArrayList	439
LinkedList	13

(Nota: El temps varia molt poc entre múltiples execucions del programa, però tampoc no és fix. En general, l'ArrayList triga entre 420 i 440 mil·lisegons, mentre la LinkedList en triga entre 10 i 15.)