Act 2.7.- Parking

Main

```java
public class Main {

    public static void main(String[] args) {
        ArrayList<Coche> arr = new ArrayList<>();
        Parking parking = new Parking();

        for (byte i = 0; i < 50; i++) {
            Coche c = new Coche(parking);
            arr.add(e: c);
            c.start();

            try {
                Thread.sleep(millis: 1500);
            } catch (InterruptedException ex) {
                System.out.printf(format: "ERROR: %s\n", args: ex.getMessage());
            }
        }

        for (byte i = 0; i < 50; i++) {
            try {
                arr.get(index: i).join();
            } catch (InterruptedException ex) {
                System.out.println(x: ex.getMessage());
            }
        }


    }

}
```

Coche

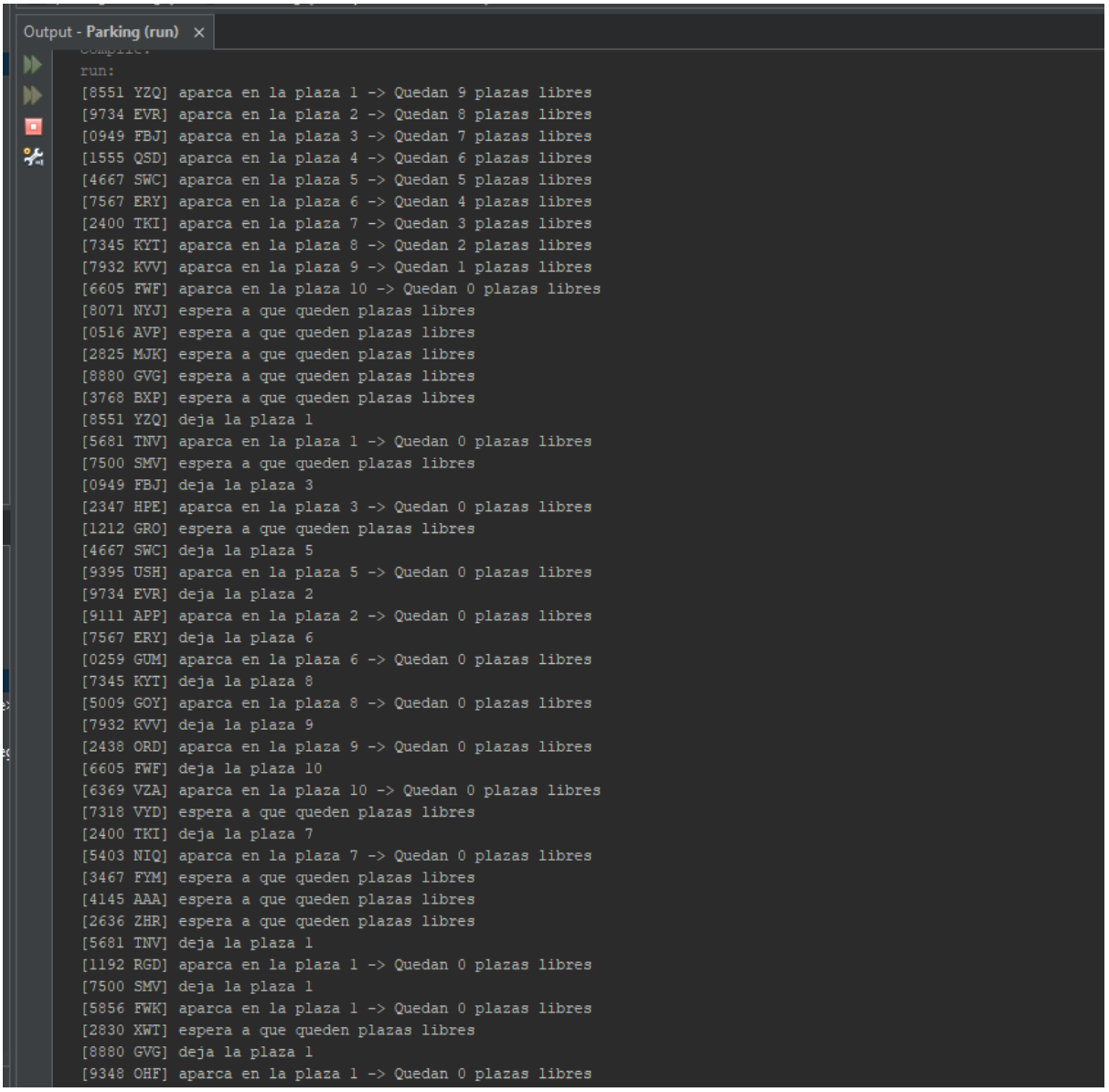
```
public class Coche extends Thread {

    private final String matricula;
    private final Parking parking;
    private int posicionParking;

    public Coche(Parking parking) {
        this.parking = parking;
        this.matricula = asignarMatricula();
        this.posicionParking = 0;
    }

    public String getMatricula() {
        return matricula;
    }

    public int getPosicionParking() {
        return posicionParking;
    }

    public void setPosicionParking(int posicionParking) {
        this.posicionParking = posicionParking;
    }

    private String asignarMatricula() {

        Random r = new Random();
        String ret = "[";
        ret += String.format(format: "%04d", args: r.nextInt(bound: 10000));
        ret += " ";

        for (byte i = 0; i < 3; i++) {
            char c = (char) ('A' + r.nextInt(bound: 26));
            ret += c;
        }
        ret += "]";
        return ret;
    }
```

```
    @Override
    public void run() {
        try {
            Random r = new Random();
            parking.accederParking(c: this);
            Thread.sleep(millis: r.nextInt(origin: 20000, bound: 30001));
            parking.salirParking(c: this);
        } catch (InterruptedException ex) {
            System.out.printf(format: "ERROR: %s\n", args: ex.getMessage());
        }
    }

}
```

Parking

```java
public class Parking {

    private final Map<Integer, Coche> aparcamientos = new HashMap<>(initialCapacity:10);
    private final LinkedList<Integer> posicionesLiberadas = new LinkedList<>();
    private int plazasLibres;

    public Parking() {
        plazasLibres = 10;
    }

    public Map<Integer, Coche> getMap() {
        return aparcamientos;
    }

    public synchronized void accederParking(Coche c) throws InterruptedException {
        while (plazasLibres == 0) {
            System.out.printf(format: "%s espera a que queden plazas libres\n", args:c.getMatricula());
            wait();
        }

        int posicion;
        if (!posicionesLiberadas.isEmpty()) {
            posicion = posicionesLiberadas.poll();
        } else {
            posicion = aparcamientos.size();
        }

        this.aparcamientos.put(key:posicion, value: c);
        c.setPosicionParking(posicionParking:posicion);
        plazasLibres--;

        System.out.printf(format: "%s aparca en la plaza %d -> Quedan %d plazas libres\n", args:c.getMatricula(), posicion + 1,
                args:plazasLibres);
    }

    public synchronized void salirParking(Coche c) throws InterruptedException {
        if (plazasLibres == 0) {
            int posicion = c.getPosicionParking();
            this.aparcamientos.remove(key:posicion);
            System.out.printf(format: "%s deja la plaza %d \n", args:c.getMatricula(), posicion + 1);
            plazasLibres++;
            posicionesLiberadas.offer(e: posicion);
            notifyAll();
        }
    }
```

Resultado


Output - Parking (run)
run:
[8551 YZQ] aparca en la plaza 1 -> Quedan 9 plazas libres
[9734 EVR] aparca en la plaza 2 -> Quedan 8 plazas libres
[0949 FBJ] aparca en la plaza 3 -> Quedan 7 plazas libres
[1555 QSD] aparca en la plaza 4 -> Quedan 6 plazas libres
[4667 SWC] aparca en la plaza 5 -> Quedan 5 plazas libres
[7567 ERY] aparca en la plaza 6 -> Quedan 4 plazas libres
[2400 TKI] aparca en la plaza 7 -> Quedan 3 plazas libres
[7345 KYT] aparca en la plaza 8 -> Quedan 2 plazas libres
[7932 KVV] aparca en la plaza 9 -> Quedan 1 plazas libres
[6605 FWF] aparca en la plaza 10 -> Quedan 0 plazas libres
[8071 NYJ] espera a que queden plazas libres
[0516 AVP] espera a que queden plazas libres
[2825 MJK] espera a que queden plazas libres
[8880 GVG] espera a que queden plazas libres
[3768 BXP] espera a que queden plazas libres
[8551 YZQ] deja la plaza 1
[5681 TNV] aparca en la plaza 1 -> Quedan 0 plazas libres
[7500 SMV] espera a que queden plazas libres
[0949 FBJ] deja la plaza 3
[2347 HPE] aparca en la plaza 3 -> Quedan 0 plazas libres
[1212 GRO] espera a que queden plazas libres
[4667 SWC] deja la plaza 5
[9395 USH] aparca en la plaza 5 -> Quedan 0 plazas libres
[9734 EVR] deja la plaza 2
[9111 APP] aparca en la plaza 2 -> Quedan 0 plazas libres
[7567 ERY] deja la plaza 6
[0259 GUM] aparca en la plaza 6 -> Quedan 0 plazas libres
[7345 KYT] deja la plaza 8
[5009 GOY] aparca en la plaza 8 -> Quedan 0 plazas libres
[7932 KVV] deja la plaza 9
[2438 ORD] aparca en la plaza 9 -> Quedan 0 plazas libres
[6605 FWF] deja la plaza 10
[6369 VZA] aparca en la plaza 10 -> Quedan 0 plazas libres
[7318 VYD] espera a que queden plazas libres
[2400 TKI] deja la plaza 7
[5403 NIQ] aparca en la plaza 7 -> Quedan 0 plazas libres
[3467 FYM] espera a que queden plazas libres
[4145 AAA] espera a que queden plazas libres
[2636 ZHR] espera a que queden plazas libres
[5681 TNV] deja la plaza 1
[1192 RGD] aparca en la plaza 1 -> Quedan 0 plazas libres
[7500 SMV] deja la plaza 1
[5856 FWK] aparca en la plaza 1 -> Quedan 0 plazas libres
[2830 XWT] espera a que queden plazas libres
[8880 GVG] deja la plaza 1
[9348 OHF] aparca en la plaza 1 -> Quedan 0 plazas libres