

## Ejercicio 1

```
Consola de depuración de Mi X + v

Valores hash de cada código:
Hash('MZ123') = 2
Hash('QK456') = 0
Hash('PL789') = 2
Hash('MN321') = 4
Hash('QK654') = 0
Hash('PL987') = 2
Hash('MZ321') = 2

Insertando códigos en la tabla hash:
Insertado 'MZ123' en posición 2
Insertado 'QK456' en posición 0
Insertado 'PL789' en posición 2
Insertado 'MN321' en posición 4
Insertado 'QK654' en posición 0
Insertado 'PL987' en posición 2
Insertado 'MZ321' en posición 2

Tabla hash resultante (con resolución de colisiones por encadenamiento):
Posición 0: QK456 -> QK654
Posición 1: (vacía)
Posición 2: MZ123 -> PL789 -> PL987 -> MZ321
Posición 3: (vacía)
Posición 4: MN321
Posición 5: (vacía)
Posición 6: (vacía)

Demostrando búsqueda de códigos:
Búsqueda de 'MZ123': Encontrado
Búsqueda de 'QK456': Encontrado
```

```
Demostrando búsqueda de códigos:
Búsqueda de 'MZ123': Encontrado
Búsqueda de 'QK456': Encontrado
Búsqueda de 'PL789': Encontrado
Búsqueda de 'MN321': Encontrado
Búsqueda de 'QK654': Encontrado
Búsqueda de 'PL987': Encontrado
Búsqueda de 'MZ321': Encontrado
Búsqueda de 'XX999': No encontrado
```

## Ejercicio 2

```
Ejecutando el algoritmo de Dijkstra desde la ciudad A hasta la ciudad E:  
La distancia más corta desde A hasta E es: 8 km  
La ruta es: A -> C -> D -> E
```

```
Detalles de la ruta:
```

```
A -> C: 2 km
```

```
C -> D: 4 km
```

```
D -> E: 2 km
```

```
|
```

### Ejercicio 3

```
Lista de pacientes (orden de llegada):
```

```
1. Ana (Prioridad 3)
```

```
2. Luis (Prioridad 1)
```

```
3. Sofía (Prioridad 2)
```

```
4. Marta (Prioridad 1)
```

```
5. Pedro (Prioridad 2)
```

```
Orden de atención según cola de prioridad mínima:
```

```
1. Luis (Prioridad 1)
```

```
2. Marta (Prioridad 1)
```

```
3. Pedro (Prioridad 2)
```

```
4. Sofía (Prioridad 2)
```

```
5. Ana (Prioridad 3)
```