Algoritmos de búsqueda

Búsqueda lineal = Búsqueda secuencial

```
// Búsqueda lineal de un elemento en un vector
// - Devuelve la posición de “dato” en el vector
// - Si “dato” no está en el vector, devuelve -1

static int buscar (double vector[], double dato)
{
  int i;
  int N = vector.length;
  int pos = -1;

  for (i=0; i<N; i++)
      if (vector[i]==dato)
         pos = i;

  return pos;
}
```

Versión mejorada

```
// Búsqueda lineal de un elemento en un vector
// - Devuelve la posición de “dato” en el vector
// - Si “dato” no está en el vector, devuelve -1

static int buscar (double vector[], double dato)
{
  int i;
  int N = vector.length;
  int pos = -1;

  for (i=0; (i<N) && (pos==-1); i++)
      if (vector[i]==dato)
         pos = i;

  return pos;
}
```

Búsqueda binaria

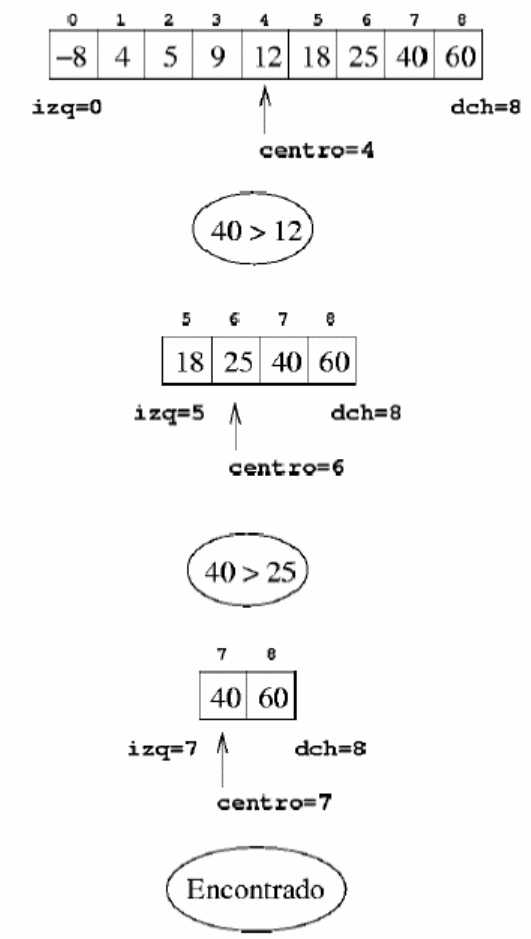
Precondición

El vector ha de estar ordenado

Algoritmo

Se compara el dato buscado
con el elemento en el centro del vector:

- Si coinciden, hemos encontrado el dato buscado.
- Si el dato es mayor que el elemento central del vector, tenemos que buscar el dato en segunda mitad del vector.
- Si el dato es menor que el elemento central del vector, tenemos que buscar el dato en la primera mitad del vector.



```
// Búsqueda binaria de un elemento en un vector
// - Devuelve la posición de “dato” en el vector
// - Si “dato” no está en el vector, devuelve -1

// Implementación recursiva
// Uso: binSearch(vector,0,vector.length-1,dato)

static int binSearch
    (double v[], int izq, int der, double buscado)
{
  int centro = (izq+der)/2;

  if (izq>der)
     return -1;
  else if (buscado==v[centro])
     return centro;
  else if (buscado<v[centro])
     return binSearch(v, izq, centro-1, buscado);
  else
     return binSearch(v, centro+1, der, buscado);
}

// Implementación iterativa
// Uso: binSearch (vector, dato)

static int binSearch (double v[], double buscado)
{
  int izq = 0;
  int der = v.length-1;
  int centro = (izq+der)/2;

  while ((izq<=der) && (v[centro]!=buscado)) {

    if (buscado<v[centro])
       der = centro – 1;
    else
       izq = centro + 1;

    centro = (izq+der)/2;
  }

  if (izq>der)
     return -1;
  else
     return centro;
}
```