



CICLO 01

[FORMACIÓN POR CICLOS]  
Fundamentos de  
**Programación**

Colección de datos  
Borrado en vector



# Operación de borrado en vector

**d = 16**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

El proceso consiste en buscar en cuál posición está el dato a borrar.

Luego se desplazan, una posición hacia la izquierda, los datos que están a continuación del dato a borrar con el fin de que no queden espacios intermedios libres en el vector.

Por último se actualiza **V[0]**.

Tendremos dos subprogramas: uno que busca el dato a borrar (este subprograma retorna la posición donde se halla el dato a borrar) y otro que efectúa el movimiento de datos en el vector.

Es importante notar que en el subprograma para buscar el dato a borrar, nuestro algoritmo retornará **-1** si no encuentra el dato buscado.

**d = 16**

**i**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

**i**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

i

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

i

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

i

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```



**d = 16**

i

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

i

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

i



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Nuestros subprogramas son:

```
def buscarDato(V, d):  
    i = 1  
    while i <= V[0] and V[i] != d:  
        i = i + 1  
    if i <= V[0]:  
        return i  
    return -1
```

**d = 16**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	16	28	5	39	44	7	12										

Diagram illustrating an array V of size 25. The array contains values: 14, 3, 1, 6, 2, 8, 4, 9, 16, 28, 5, 39, 44, 7, 12, followed by 10 empty slots. A blue arrow labeled 'i' points to the element 16 at index 8. An orange arrow labeled 'j' points to the element 16 at index 8.

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```

**d = 16**

	0	1	2	3	4	5	6	7	i	9	j	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
V	14	3	1	6	2	8	4	9	28	28	5	39	44	7	12											

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```

**d = 16**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
<b>V</b>	14	3	1	6	2	8	4	9	28	5	5	39	44	7	12										

*i* points to index 8, *j* points to index 10

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```

**d = 16**

	0	1	2	3	4	5	6	7	i	9	10	j	12	13	14	15	16	17	18	19	20	21	22	23	24
V	14	3	1	6	2	8	4	9	28	5	39	39	44	7	12										

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```

**d = 16**

	0	1	2	3	4	5	6	7	i	9	10	j	12	13	14	15	16	17	18	19	20	21	22	23	24
V	14	3	1	6	2	8	4	9	28	5	39	44	44	7	12										

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```



**d = 16**

	0	1	2	3	4	5	6	7	i	9	10	j	12	13	14	15	16	17	18	19	20	21	22	23	24
V	14	3	1	6	2	8	4	9	28	5	39	44	7	7	12										

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```

**d = 16**

	0	1	2	3	4	5	6	7	i 8	9	10	11	12	13	j 14	15	16	17	18	19	20	21	22	23	24
V	14	3	1	6	2	8	4	9	28	5	39	44	7	12	12										

Nuestros subprogramas son:

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```

**d = 16**

i



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
V	13	3	1	6	2	8	4	9	28	5	39	44	7	12											

```
def borrar(V, i):  
    for j in range(i, V[0]):  
        V[j] = V[j + 1]  
    V[0] = V[0] - 1
```