# Passo 1:

Declaração e inicialização da matriz no programa principal:

int[] matriz = { 23, -9, 13, 9 };

# Passo 2:

Chamada da função bubbleSort() passando a matriz como argumento:

bubbleSort(matriz, 4);

# Passo 3:

Execução da função bubbleSort():

void bubbleSort(int \*arr, int size) {

int i, j;

for (i = 0; i < size - 1; i++) {

for (j = 0; j < size - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

swap(&arr[j], &arr[j + 1]);

}

}

}

}

# Passo 4:

Iteração externa do bubble sort (i = 0):

i = 0, j = 0: Comparando arr[0] = 23 e arr[1] = -9 (23 > -9? Sim)

Realizando swap(arr[0], arr[1])

A matriz agora é: { -9, 23, 13, 9 }

# Passo 5:

Iteração interna do bubble sort (j = 1):

i = 0, j = 1: Comparando arr[1] = 23 e arr[2] = 13 (23 > 13? Sim)

Realizando swap(arr[1], arr[2])

A matriz agora é: { -9, 13, 23, 9 }

# Passo 6:

Iteração interna do bubble sort (j = 2):

i = 0, j = 2: Comparando arr[2] = 23 e arr[3] = 9 (23 > 9? Sim)

Realizando swap(arr[2], arr[3])

A matriz agora é: { -9, 13, 9, 23 }

# Passo 7:

Iteração externa do bubble sort (i = 1):

i = 1, j = 0: Comparando arr[0] = -9 e arr[1] = 13 (-9 > 13? Não)

Nenhuma troca é realizada

A matriz permanece como: { -9, 13, 9, 23 }

# Passo 8:

Iteração interna do bubble sort (j = 1):

i = 1, j = 1: Comparando arr[1] = 13 e arr[2] = 9 (13 > 9? Sim)

Realizando swap(arr[1], arr[2])

A matriz agora é: { -9, 9, 13, 23 }

# Passo 9:

Iteração externa do bubble sort (i = 2):

i = 2, j = 0: Comparando arr[0] = -9 e arr[1] = 9 (-9 > 9? Não)

Nenhuma troca é realizada

A matriz permanece como: { -9, 9, 13, 23 }

# Passo 10:

Iteração externa do bubble sort (i = 3):

i = 3, j = 0: Comparando arr[0] = -9 e arr[1] = 9 (-9 > 9? Não)

Nenhuma troca é realizada

A matriz permanece como: { -9, 9, 13, 23 }

# Passo 11:

Fim do bubble sort. A matriz está ordenada:

A matriz final é: { -9, 9, 13, 23 }