1. What is the time complexity of the Bubble Sort algorithm in the worst case?
   A. O(n)
   B. O(n log n)
   C. O(n^2)
   D. O(2^n)

2. What happens to the largest element in the array after the first complete iteration of Bubble Sort?
   A. It moves to the first position
   B. It remains in its original position
   C. It moves to the last position
   D. Its position is randomized
3. Which of the following scenarios can Bubble Sort handle efficiently?
   A. Sorting very large datasets
   B. Sorting nearly sorted datasets
   C. Sorting datasets with a lot of duplicate values
   D. Sorting datasets with complex custom compare functions
4. In the Bubble Sort algorithm, how many comparisons does the outer loop typically perform for an array of size 'n'?
   A. n
   B. n-1
   C. n^2
   D. n/2
5. For the given bubble sort algorithm:

If we want to sort the elements [5,56,3,65,2,1], what should be the value of blank?

```
def bubble_sort(elements):
    size = len(elements)

    for i in range(Blank):
        swapped = False
        for j in range(size-1-i):
            if elements[j] > elements[j+1]:
                tmp = elements[j]
                elements[j] = elements[j+1]
                elements[j+1] = tmp
                swapped = True

        if not swapped:
            break
```

A. size-1-i
B. Size-1
C. size
D. 1