1. Explanation of Main Sorting Algorithms with Examples:

a. Bubble Sort:

Bubble Sort is a simple comparison-based sorting algorithm that iterates through the list to be sorted multiple times, comparing neighboring items and swapping them if they're out of order. The list step is repeated until the entire list is sorted. Example of Bubble Sort:
Let's consider an array: [5, 2, 9, 1, 5, 6]. The steps of the Bubble Sort would be as follows:
1. [**5, 2**, 9, 1, 5, 6] (Swap 5 and 2)
2. [5, **2, 9**, 1, 5, 6] (Swap 9 and 2)
3. [5, 2, **9, 1**, 5, 6] (Swap 9 and 1)
4. [5, 2, 1, **9, 5**, 6] (Swap 9 and 5)
5. [5, 2, 1, 5, **9, 6**] (Swap 9 and 6)
6. [**5, 2**, 1, 5, 6, 9] (Swap 5 and 2)
7. [2, **5, 1**, 5, 6, 9] (Swap 5 and 1)
8. [2, 1, **5, 5**, 6, 9] (Swap 5 and 5)

The sorted array: [1, 2, 5, 5, 6, 9].

b. Selection Sort:

Selection Sort is a basic sorting algorithm that divides the list into two parts: the sorted part and the unsorted part. It iteratively finds the smallest element in the unsorted part and swaps it with the first element of the unsorted part until the entire list is sorted. Let’s consider an array: [8, 3, 1, 7, 5, 2]. The steps of the Selection Sort would be as follows:

1. [**1**, 3, 8, 7, 5, 2] (Swap 1 and 8)
2. [1, **2**, 8, 7, 5, 3] (Swap 2 and 3)
3. [1, 2, **3**, 7, 5, 8] (Swap 3 and 8)
4. [1, 2, 3, **5**, 7, 8] (Swap 5 and 8)
5. [1, 2, 3, 5, **7**, 8] (Swap 7 and 8)

The sorted array: [1, 2, 3, 5, 7, 8].

2. Comparison of Bubble Sort and Selection Sort:
Basic sorting algorithms having an average time complexity of O(n2) include Bubble Sort and Selection Sort. However, their methods and levels of effectiveness vary:

Bubble Sort: It swaps neighboring elements if they are in the wrong order by comparing adjacent elements.
- The method must traverse around the array several times, shifting the biggest element to the right place with each pass.
- For sorting huge datasets, it is not the most effective option because to its time complexity of O(n2).

Selection Sort: - It locates the lowest element in the unsorted section and replaces it with the unsorted part's first element.
- In each iteration, the method only needs to make one trip through the array to determine the minimum element.

3. Real-World Examples of Sorting:
Sorting is a typical operation in a variety of real-world situations, such as:

- Sorting product lists on e-commerce websites according to relevance, popularity, or price.
- Sorting files into a file system to make them easier to find and access quickly.
- Listing search results according to their relevance or date of publication.
- Classifying student grades to rank them in a classroom.
- Setting up consumer data in a database for effective analysis and querying.
- Classifying projects or to-do lists according to priority or deadline.

4. Sorting an Integer Array using Bubble Sort and Selection Sort (Python):

PYTHON

```
# Bubble Sort
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

# Selection Sort
def selection_sort(arr):
    n = len(arr)
    for i in range(n):
        min_idx = i
        for j in range(i + 1, n):
            if arr[j] < arr[min_idx]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]

# Example usage:
arr = [64, 34, 25, 12, 22, 11, 90]
print("Original Array:", arr)

bubble_sort(arr)
print("Array sorted using Bubble Sort:", arr)

arr = [64, 34, 25, 12, 22, 11, 90]
selection_sort(arr)
print("Array sorted using Selection Sort:", arr)
```

Although Selection Sort and Bubble Sort are simple to comprehend, they are not the most effective sorting techniques. Consider utilizing more complex sorting algorithms like Merge Sort or Quick Sort for larger datasets.