Hello

Hi there! How's it going?

Help me implement three classic sorting algorithms in python

Sure! Here are implementations for three classic sorting algorithms in Python: Bubble Sort, Insertion Sort, and Quick Sort.

1. Bubble Sort

Bubble Sort repeatedly compares adjacent elements and swaps them if they are in the wrong order. It continues this process until no swaps are needed.

python 复制代码

```python
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]
    return arr

# Example usage
arr = [64, 34, 25, 12, 22, 11, 90]
print("Bubble Sort:", bubble_sort(arr))
```

2. Insertion Sort

Insertion Sort builds the final sorted array one item at a time. It picks the next element and inserts it into the correct position in the already sorted portion of the array.

```python
def insertion_sort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i-1
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
    return arr

# Example usage
arr = [64, 34, 25, 12, 22, 11, 90]
print("Insertion Sort:", insertion_sort(arr))
```

3. Quick Sort

Quick Sort is a divide-and-conquer algorithm. It picks an element as a pivot and partitions the array around the pivot, then recursively sorts the subarrays.

```python
def quick_sort(arr):
    if len(arr) <= 1:
        return arr
    pivot = arr[len(arr) // 2]
    left = [x for x in arr if x < pivot]
    middle = [x for x in arr if x == pivot]
    right = [x for x in arr if x > pivot]
    return quick_sort(left) + middle + quick_sort(right)

# Example usage
arr = [64, 34, 25, 12, 22, 11, 90]
print("Quick Sort:", quick_sort(arr))
```

Feel free to try these out and let me know if you have any questions or need further assistance!