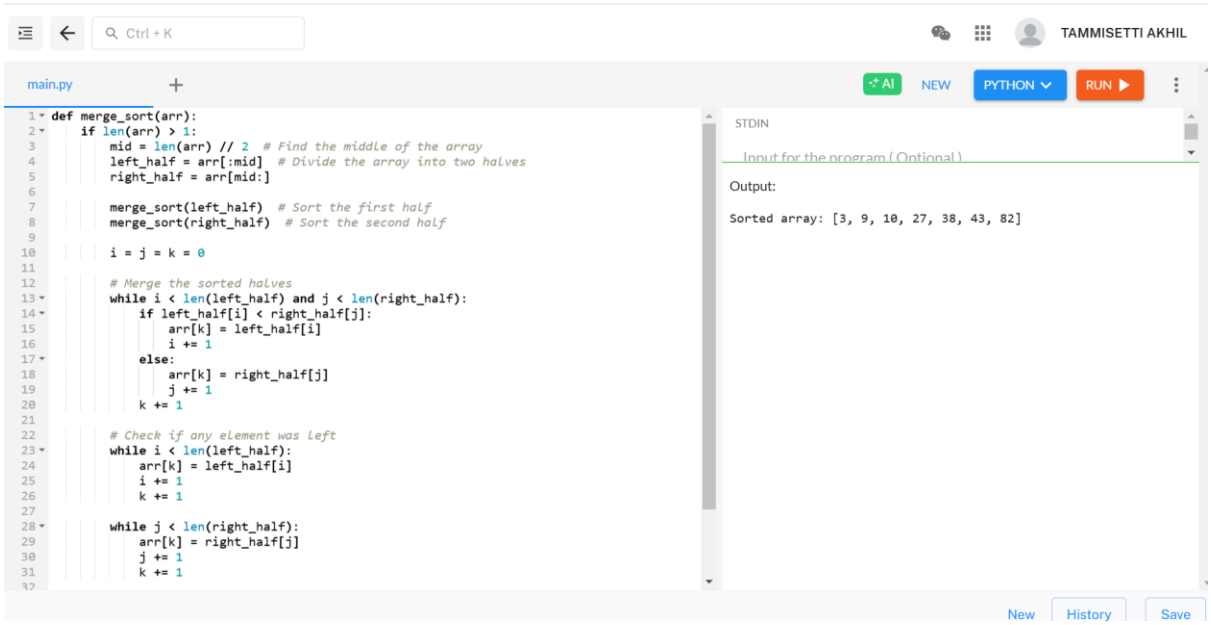
```
def merge_sort(arr):
    if len(arr) > 1:
        mid = len(arr) // 2  # Find the middle of the array
        left_half = arr[:mid]  # Divide the array into two halves
        right_half = arr[mid:]
        merge_sort(left_half)  # Sort the first half
        merge_sort(right_half)  # Sort the second half
        i = j = k = 0
        # Merge the sorted halves
        while i < len(left_half) and j < len(right_half):
            if left_half[i] < right_half[j]:
                arr[k] = left_half[i]
                i += 1
            else:
                arr[k] = right_half[j]
                j += 1
            k += 1
        # Check if any element was left
        while i < len(left_half):
            arr[k] = left_half[i]
            i += 1
            k += 1
        while j < len(right_half):
            arr[k] = right_half[j]
            j += 1
            k += 1
# Example usage
arr = [38, 27, 43, 3, 9, 82, 10]
merge_sort(arr)
print("Sorted array:", arr)
```




```python
def merge_sort(arr):
    if len(arr) > 1:
        mid = len(arr) // 2  # Find the middle of the array
        left_half = arr[:mid]  # Divide the array into two halves
        right_half = arr[mid:]

        merge_sort(left_half)  # Sort the first half
        merge_sort(right_half)  # Sort the second half

        i = j = k = 0

        # Merge the sorted halves
        while i < len(left_half) and j < len(right_half):
            if left_half[i] < right_half[j]:
                arr[k] = left_half[i]
                i += 1
            else:
                arr[k] = right_half[j]
                j += 1
            k += 1

        # Check if any element was left
        while i < len(left_half):
            arr[k] = left_half[i]
            i += 1
            k += 1

        while j < len(right_half):
            arr[k] = right_half[j]
            j += 1
            k += 1
```