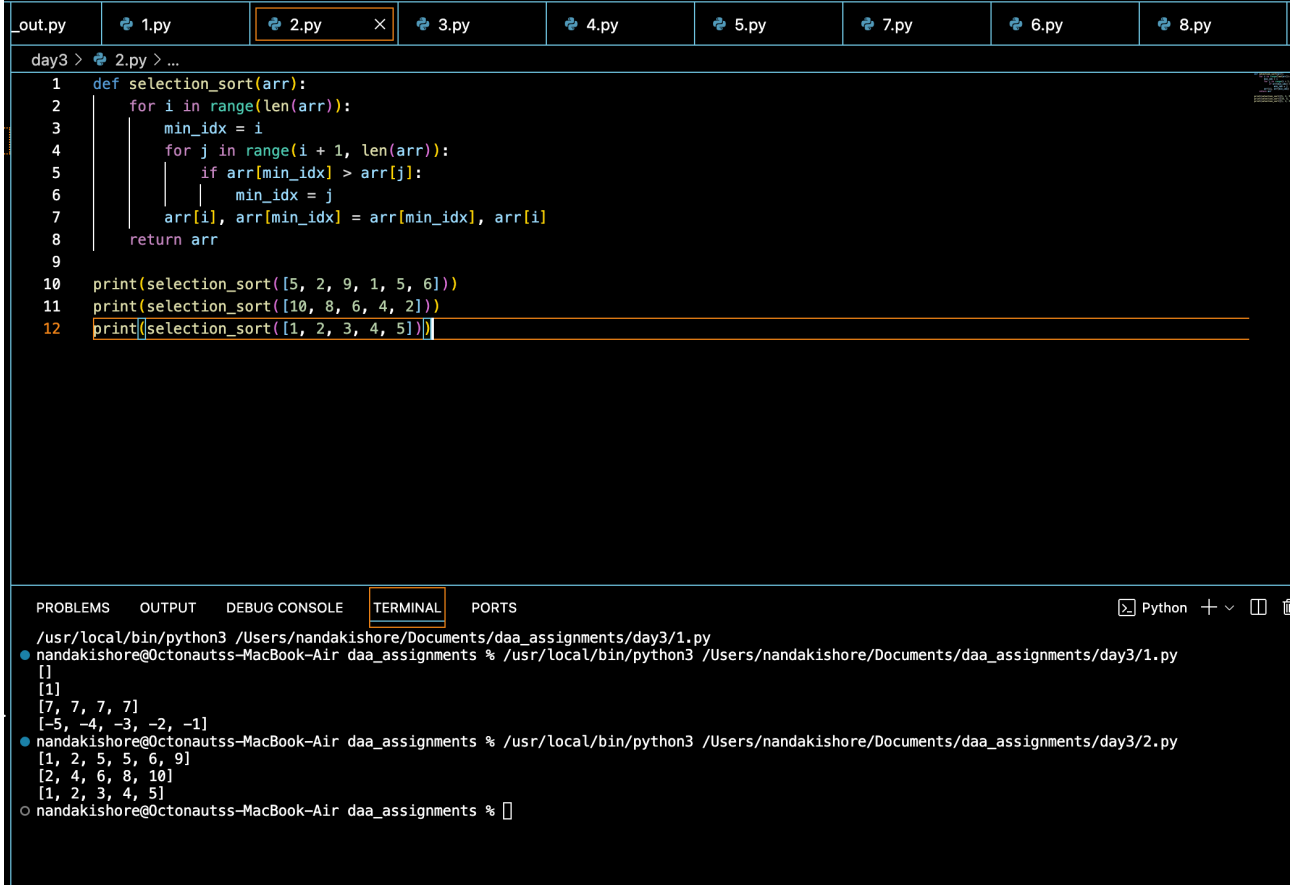
1.

```
def perform_on_list lst
    return sorted lst

print perform_on_list
print perform_on_list  1
print perform_on_list  7  7  7  7
print perform_on_list  -5  -1  -3  -2  -4
```

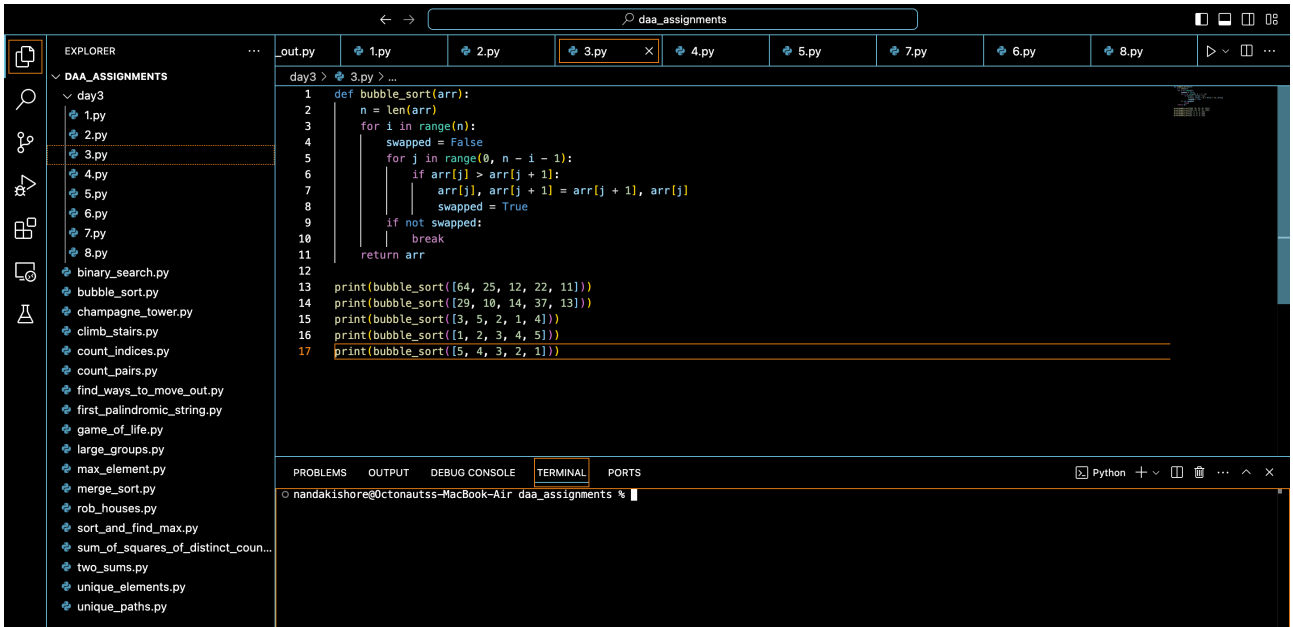
2.

```
def selection_sort arr
    for i in range len arr
        min_idx = i
        for j in range(i + 1, len(arr)):
            if arr[min_idx] > arr[j]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]
    return arr

print selection_sort  5  2  9  1  5  6
print selection_sort  10  8  6  4  2
print selection_sort  1  2  3  4  5
```



3.

```
def bubble_sort arr
    n = len(arr)
    for i in range n
        swapped = False
        for j in range(0, n - i - 1):
            if arr[] > arr[j + 1]:
                arr[], arr[j + 1] = arr[j + 1], arr[]
                swapped = True
        if not swapped:
            break
    return arr

print bubble_sort  64  25  12  22  11
print bubble_sort  29  10  14  37  13
print bubble_sort  3  5  2  1  4
print bubble_sort  1  2  3  4  5
print bubble_sort  5  4  3  2  1
```



4.

```
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

print insertion_sort  3  1  4  1  5  9  2  6  5  3
print insertion_sort  5  5  5  5  5
print insertion_sort  2  3  1  3  2  1  1  3
```

5.

```
def find_kth_missing_positive arr  k
    missing =
    current = 1
    i = 0
    while len missing  < k
        if i < len(arr) and arr[i] == current:
            i += 1
        else:
            missing.append(current)
```

```
        current += 1
    return missing - 1

print find_kth_missing_positive 2 3 4 7 11 5
print find_kth_missing_positive 1 2 3 4 2
```

6.

```
def find_peak_element nums
    left  right = 0  len nums  - 1
    while left < right
        mid = (left + right) // 2
        if nums[mid] > nums[mid + 1]:
            right = mid
        else:
            left = mid + 1
    return left

print find_peak_element 1 2 3 1
print find_peak_element 1 2 1 3 5 6 4
```

7.

```
def str_str haystack  needle
    return haystack        needle

print str_str "sadbutsad"  "sad"
print str_str "leetcode"  “leeto"
```

8.

```
def find_substrings words
    result =
    for i  word in enumerate words
        for j  other in enumerate words
            if i != j and word in other:
                result.append(word)
                break
    return result

print find_substrings  "mass"  "as"  "hero"  "superhero"
print find_substrings  "leetcode"  "et"  "code"
print find_substrings  "blue"  "green"  "bu"
```