SWE30009- Software Testing and Reliability

Assignment 2

Name: Pham Do Tien Phong

Student ID: 104189767

Task 1:

Test case 1: General cases with duplicated integers

- Input: [6, 1, 3, 4, 3, 5, 1, 19, 17, 18, 11, 13, 11, 12, 14, 20, 1, 22]
- Output: [1, 3, 5, 11, 13, 17, 19] and [4, 6, 12, 14, 18, 20, 22]
- Objectives: Since this test case includes duplicated integers, it may be used to evaluate how well duplicated integers are removed.

Test case 2: All even integers

- Input: [2, 50, 12 , 8, 6]
- Output:[] and [2, 6, 8, 12, 50]
- Objectives: This test case serve to test the program‘s handling of a list containing only even integers and create an empty list with no odd integers

Test case 3: All odd integers

- Input:[15, 91, 83, 11, 7, 13, 21, 65]
- Output: [7, 11, 13, 15, 21, 65, 83, 91] and []
- Objectives: This test case implements the scenario that the output list only contains odd integers and an empty list of even integers.

Test case 4: General test case with duplicated integers and largest valid input size

- Input: [22, 1, 13, 14, 1, 22, 3, 2, 5, 1, 8, 9, 1, 2, 5, 5, 1, 4, 3]
- Output: [1, 3, 5, 9, 13] and [2, 4, 8, 14, 22]
- Objectives: This test case serve to test the program ‘s handling of the largest possible valid input list and 2 output lists (an odd integers list and an even integers list) without duplicate lists

Test case 5: Maximum input size

- Input: [22, 1, 13, 14, 1, 22, 3, 2, 5, 1, 8, 9, 1, 2, 5, 5, 1, 4, 3, 1, 3, 15, 4, 1, 2]
- Output: Error or rejection input list
- Objectives: This test case is used to test the maximum valid input list size .

Test case 6: 0 in input list

- Input: [3, 5, 0]
- Output: Error or rejection input list
- Objectives: This test case serve to act the error message when 0 is a integer in input list.

Task 2:

My choice of single test case among the 6 test cases proposed above must be test case 4. The test case adapts almost all of the testing objectives for the program:

- This test case serves to test the program ‘s handling of the largest possible valid input list and 2 output lists without duplicated integers
- The list contains both odd and even integers
- The list contains duplicated integers that must be removed
- The list is long, which means the test for maximum input list size is adapted

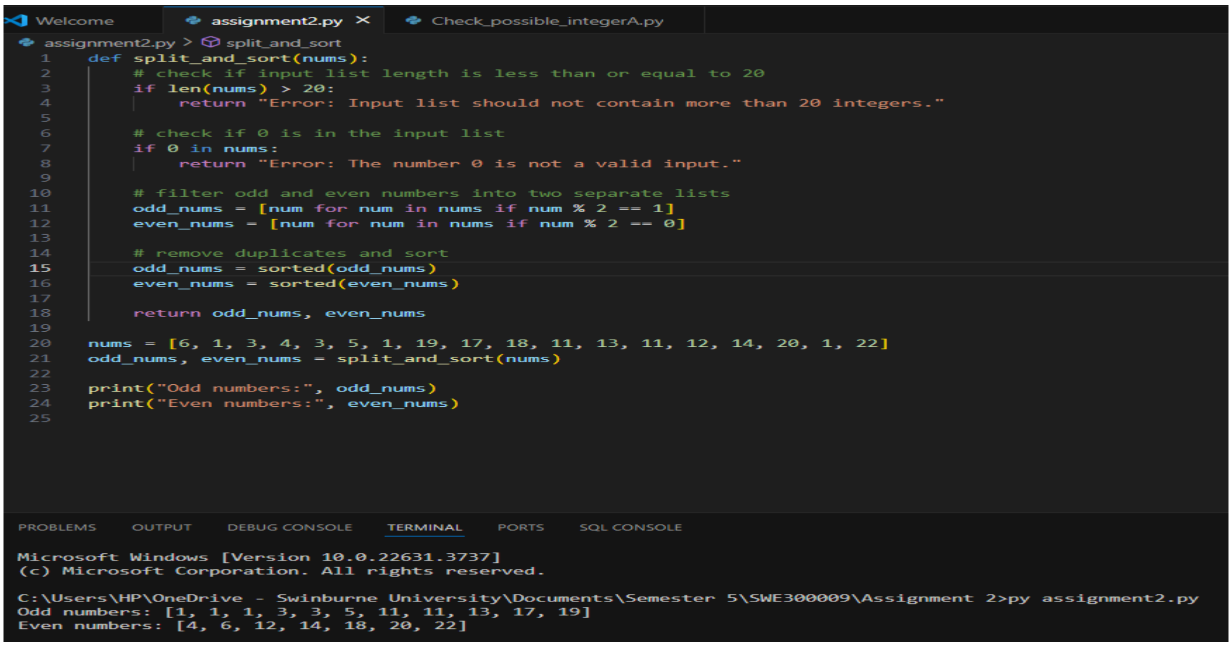
Explanation and justification:

1. Complexity and coverage:
    - Test Case 4 covers the handling of duplicates by including multiple instances of several numbers.
    - It tests the program’s ability to sort and remove duplicates in both odd and even lists.
    - The length of the input list tests the program’s ability to handle the maximum input size constraint.
2. Realistic scenario:
    - This test case simulates a realistic and common scenario where input data may have duplicates and a mix of odd and even integers.
    - It ensures that both output lists (odd and even) are non-empty, thus testing the core functionality of the program thoroughly.
3. Productivity:
    - By including multiple duplicates and having a length close to the maximum limit, it tests the program’s robustness and efficiency.
    - It does not include any invalid inputs like 0, focusing on valid edge cases, which are more common in real-world applications.

Task 3:

1. Suggestions for Improvement

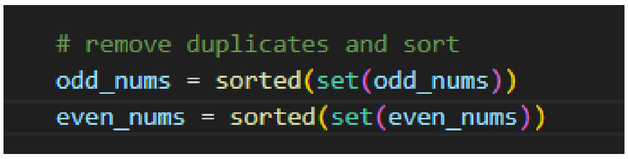
The first necessary improvement for the program must be implemented in the removal of duplicated integers due to the fact that the initial program provided only sort but the duplicated integers still remain



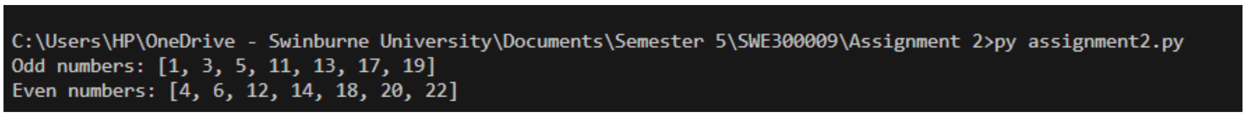
```python
def split_and_sort(nums):
    # check if input list length is less than or equal to 20
    if len(nums) > 20:
        return "Error: Input list should not contain more than 20 integers."

    # check if 0 is in the input list
    if 0 in nums:
        return "Error: The number 0 is not a valid input."

    # filter odd and even numbers into two separate lists
    odd_nums = [num for num in nums if num % 2 == 1]
    even_nums = [num for num in nums if num % 2 == 0]

    # remove duplicates and sort
    odd_nums = sorted(odd_nums)
    even_nums = sorted(even_nums)

    return odd_nums, even_nums

nums = [6, 1, 3, 4, 3, 5, 1, 19, 17, 18, 11, 13, 11, 12, 14, 20, 1, 22]
odd_nums, even_nums = split_and_sort(nums)

print("Odd numbers:", odd_nums)
print("Even numbers:", even_nums)
```

```
Microsoft Windows [Version 10.0.22631.3737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py
Odd numbers: [1, 1, 1, 3, 3, 5, 11, 11, 13, 17, 19]
Even numbers: [4, 6, 12, 14, 18, 20, 22]
```

In output list, the duplicated integers, such as 1, 3, and 11, are not removed. It is an important bug, as it can not adapt to the requirement of the program.



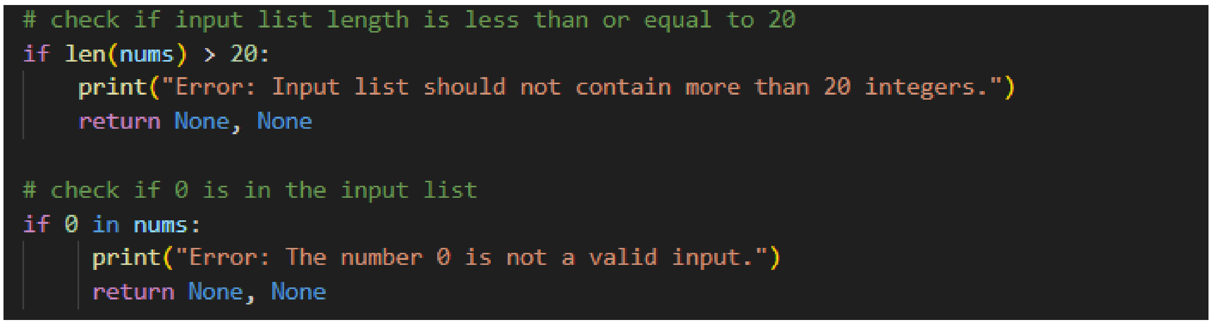
```python
# remove duplicates and sort
odd_nums = sorted(set(odd_nums))
even_nums = sorted(set(even_nums))
```



```
C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py
Odd numbers: [1, 3, 5, 11, 13, 17, 19]
Even numbers: [4, 6, 12, 14, 18, 20, 22]
```

With the change in odd_nums and even_nums operations, this leads to the removal of duplicated integers, adapting the accuracy of the program.

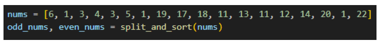
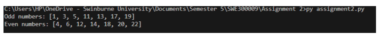
The second suggestion for the development of this program is to add two printed error messages whenever there are invalid inputs.

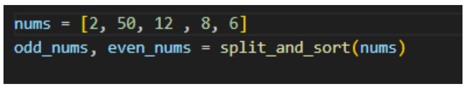
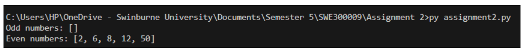


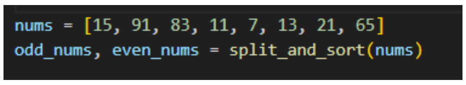
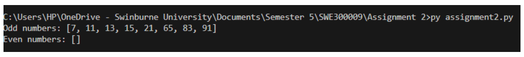
```python
# check if input list length is less than or equal to 20
if len(nums) > 20:
    print("Error: Input list should not contain more than 20 integers.")
    return None, None

# check if 0 is in the input list
if 0 in nums:
    print("Error: The number 0 is not a valid input.")
    return None, None
```

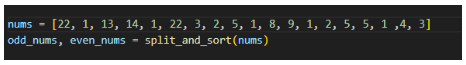
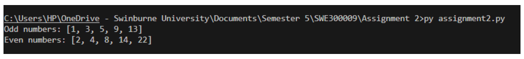
With this addition, the testers will be easier to operate as well as debug if needed.

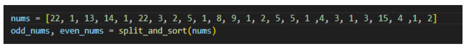
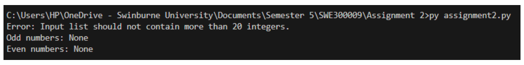
2. Report the individual test cases and the results:

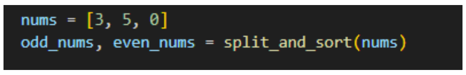
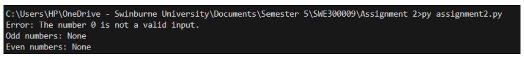
Test case	Output
 nums = [6, 1, 3, 4, 3, 5, 1, 19, 17, 18, 11, 13, 11, 12, 14, 20, 1, 22] odd_nums, even_nums = split_and_sort(nums)	 C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py Odd numbers: [1, 3, 5, 11, 13, 17, 19] Even numbers: [4, 6, 12, 14, 18, 20, 22]

 `nums = [2, 50, 12 , 8, 6]`
`odd_nums, even_nums = split_and_sort(nums)`
	 `C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py`
	`Odd numbers: []`
	`Even numbers: [2, 6, 8, 12, 50]`

 `nums = [15, 91, 83, 11, 7, 13, 21, 65]`
`odd_nums, even_nums = split_and_sort(nums)`
	 `C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py`
	`Odd numbers: [7, 11, 13, 15, 21, 65, 83, 91]`
	`Even numbers: []`

 `nums = [22, 1, 13, 14, 1, 22, 3, 2, 5, 1, 8, 9, 1, 2, 5, 5, 1 ,4, 3]`
`odd_nums, even_nums = split_and_sort(nums)`
	 `C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py`
	`Odd numbers: [1, 3, 5, 9, 13]`
	`Even numbers: [2, 4, 8, 14, 22]`

 `nums = [22, 1, 13, 14, 1, 22, 3, 2, 5, 1, 8, 9, 1, 2, 5, 5, 1 ,4, 3, 1, 3, 15, 4 ,1, 2]`
`odd_nums, even_nums = split_and_sort(nums)`
	 `C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py`
	`Error: Input list should not contain more than 20 integers.`
	`Odd numbers: None`
	`Even numbers: None`

 `nums = [3, 5, 0]`
`odd_nums, even_nums = split_and_sort(nums)`
	 `C:\Users\HP\OneDrive - Swinburne University\Documents\Semester 5\SWE300009\Assignment 2>py assignment2.py`
	`Error: The number 0 is not a valid input.`
	`Odd numbers: None`
	`Even numbers: None`

3. Insights:

Error Handling:

- The results for Test Cases 5 and 6 show that the error handling for both the list size and the presence of 0 functions as intended.

Duplication Removal:

- The outcomes of Test Cases 1, 2, 3, and 4 demonstrate that the function effectively sorts the integers and eliminates duplicates.

Edge Cases:

- The function correctly handles edge cases, such as lists with only even numbers (Test Case 2) or only odd numbers (Test Case 3).