```python
# SE 113 LAB 10 SAMPLE SOLUTION
# filename: lab10.py

# LAB 10 Question 1-a Starts Here

numbers = dict()
for x in range(1, 31):
    numbers[x] = x**2
print(numbers)

print()

# LAB 10 Question 1-a Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 1-b Starts Here

for key, value in numbers.items():
    print(str(key) + ': ' + str(value))

print()

# LAB 10 Question 1-b Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 1-c Starts Here

sum_of_numbers = 0
for key, value in numbers.items():
    sum_of_numbers = sum_of_numbers + value
print('Sum of the numbers in the dictionary is ' + str(sum_of_numbers))

print()

# LAB 10 Question 1-c Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 1-d Starts Here

if 10 in numbers:
    del numbers[10]
print(numbers)

# LAB 10 Question 1-d Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 2 Starts Here

dictionary1 = {'Tony': 41, 'Steve': 39, 'Nat': 35}
dictionary2 = {'Tony': 42, 'Clint': 35, 'Thor': 38}

# LAB 10 Question 2-a Starts Here

merged_dictionary = dictionary1.copy()
merged_dictionary.update(dictionary2)

print()

print('Name \tAge')
for name, age in merged_dictionary.items():
    print('{} \t{}'.format(name, age))

print()

# LAB 10 Question 2-a Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 2-b Starts Here

if 'Nat' in merged_dictionary:
    del merged_dictionary['Nat']
print(merged_dictionary)

print()

# LAB 10 Question 2-b Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 2-c Starts Here

lst = list(merged_dictionary.keys())
lst.sort()
for key in lst:
    print(key, merged_dictionary[key])

print()

# LAB 10 Question 2-c Ends Here
# ------------------------------------------------------------------------------

# LAB 10 Question 2-d Starts Here

largest = None
for itervar in merged_dictionary.values():
    if largest is None or itervar > largest:
        largest = itervar
print('Maximum value:', largest)

smallest = None
for itervar in merged_dictionary.values():
    if smallest is None or itervar < smallest:
        smallest = itervar
print('Minimum value:', smallest)

# Alternative solution without loops
# max_key = max(merged_dictionary.keys(), key=(lambda k: merged_dictionary[k]))
# min_key = min(merged_dictionary.keys(), key=(lambda k: merged_dictionary[k]))
# print('Maximum Value: ', merged_dictionary[max_key])
# print('Minimum Value: ', merged_dictionary[min_key])

print()

# LAB 10 Question 2-d Ends Here
# ------------------------------------------------------------------------------

# LAB 10 TODO@HOME Starts Here


def func(dictionary):
    result_d = dict()
    for v in dictionary.values():  # for each value in the dictionary
        keys_of_v = list()
        for key, val in dictionary.items():
            if v == val:
                keys_of_v.append(key)  # all keys associated to same value are added into a list
                result_d[v] = keys_of_v  # this list is stored in another dictionary
    for k in result_d:
        if len(result_d[k]) > 1:
            print('The following keys are mapped to the same value,', k)
            print(result_d[k])


""" an alternative way using set data type:
def func(d):
    # Create an empty dictionary to check for duplicates.
    dict_b = {}
    # Copy all the elements of input dictionary to dict_b but group them with their values.
    # That is when dict_b is printed, the result will look like the following
    # {41: {'Tony', 'Bruce'}, 39: {'Steve'}, 35: {'Clint'}, 38: {'Thor'}}
    for key, value in d.items():
        dict_b.setdefault(value, set()).add(key)

    # The filter() function in Python takes in a function and a list as arguments.
    # It allows to filter out all the elements of a sequence, for which the function returns True.
    # lambda is an anonymous function here. When the length of k is greater than 1 that means
    # there are more than 1 key for the same value and return those keys.
    result = filter(lambda k: len(k) > 1, dict_b.values())
    result = list(result)

    # Print the result.
    for x in range(len(result)):
        print('The following keys have the same value in d:')
        print(result[x])
"""


# Calling the function with merged_dictionary here.
func(merged_dictionary)

# LAB 10 TODO@HOME Ends Here
# ------------------------------------------------------------------------------
```