Question 1
Create a function that takes a number as an argument and returns True or False depending on whether the number is symmetrical or not. A number is symmetrical when it is the same as
its reverse.
Examples
is_symmetrical(7227) ➞ True
is_symmetrical(12567) ➞ False
is_symmetrical(44444444) ➞ True
is_symmetrical(9939) ➞ False
is_symmetrical(1112111) ➞ True

Question 2
Given a string of numbers separated by a comma and space, return the product of the numbers.
Examples
multiply_nums(&quot;2, 3&quot;) ➞ 6
multiply_nums(&quot;1, 2, 3, 4&quot;) ➞ 24
multiply_nums(&quot;54, 75, 453, 0&quot;) ➞ 0
multiply_nums(&quot;10, -2&quot;) ➞ -20

Question 3
Create a function that squares every digit of a number.
Examples
square_digits(9119) ➞ 811181
square_digits(2483) ➞ 416649
square_digits(3212) ➞ 9414
Notes
The function receives an integer and must return an integer.

Question 4
Create a function that sorts a list and removes all duplicate items from it.
Examples
setify([1, 3, 3, 5, 5]) ➞ [1, 3, 5]
setify([4, 4, 4, 4]) ➞ [4]
setify([5, 7, 8, 9, 10, 15]) ➞ [5, 7, 8, 9, 10, 15]
setify([3, 3, 3, 2, 1]) ➞ [1, 2, 3]

Question 5
Create a function that returns the mean of all digits.
Examples
mean(42) ➞ 3
mean(12345) ➞ 3
mean(666) ➞ 6
Notes
The mean of all digits is the sum of digits / how many digits there are (e.g. mean of digits in 512 is (5+1+2)/3(number of digits) = 8/3=2).
The mean will always be an integer.

Task 1:

To solve this task, we need to convert the given number to a string, reverse it, and compare it with the original number. If they are the same, the number is symmetrical; otherwise, it is not.

python

```
def is_symmetrical(num):
    num_str = str(num)
    reverse_str = num_str[::-1]
    if num_str == reverse_str:
        return True
    else:
        return False
```

Let's test our function:

python

```
print(is_symmetrical(7227))   # True
print(is_symmetrical(12567))  # False
print(is_symmetrical(44444444))  # True
print(is_symmetrical(9939))   # False
print(is_symmetrical(1112111))  # True
```

Task 2:

We need to split the input string by a comma and space and then multiply all the numbers in the resulting list. We can use a for loop or the built-in `reduce()` function from the `functools` module to do this.

python

```
from functools import reduce

def multiply_nums(nums):
    num_list = nums.split(", ")
    num_list = [int(num) for num in num_list]
    product = reduce(lambda x, y: x * y, num_list)
    return product
```

Let's test our function:

python

```
print(multiply_nums("2, 3"))   # 6
print(multiply_nums("1, 2, 3, 4"))  # 24
print(multiply_nums("54, 75, 453, 0"))  # 0
print(multiply_nums("10, -2"))   # -20
```

Task 3:

We need to extract each digit from the given number, square it, and then join them back together to form a new number.

python

```
def square_digits(num):
    num_str = str(num)
    square_str = ""
    for digit in num_str:
        square = int(digit) ** 2
        square_str += str(square)
    return int(square_str)
```

Let's test our function:

python

```
print(square_digits(9119))   # 811181
print(square_digits(2483))   # 416649
print(square_digits(3212))   # 9414
```

Task 4:

We can use the built-in `set()` function to remove duplicates from the list and then convert it back to a list to get the sorted, unique elements.

python

```
def setify(lst):
    unique_lst = list(set(lst))
    unique_lst.sort()
    return unique_lst
```

Let's test our function:

python

```
print(setify([1, 3, 3, 5, 5]))   # [1, 3, 5]
print(setify([4, 4, 4, 4]))   # [4]
print(setify([5, 7, 8, 9, 10, 15]))   # [5, 7, 8, 9, 10, 15]
print(setify([3, 3, 3, 2, 1]))   # [1, 2, 3]
```

Task 5:

We need to extract each digit from the given number, sum them up, and then divide the sum by the number of digits.

python

```
def mean(num):
    num_str = str(num)
    digit_sum = sum([int(digit) for digit in num_str])
    mean = digit_sum // len(num_str)
```