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

Ans.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def is\_symmetrical(number):

if str(number) == str(number)[::-1]:

print(True)

return True

else:

print(False)

return False

try:

is\_symmetrical(7227)

is\_symmetrical(12567)

is\_symmetrical(44444444)

is\_symmetrical(9939)

is\_symmetrical(1112111)

lg.info("""Class is\_symmetrical() has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

Question 2

Given a string of numbers separated by a comma and space, return the product of the numbers.

### Examples

multiply\_nums("2, 3") ➞ 6

multiply\_nums("1, 2, 3, 4") ➞ 24

multiply\_nums("54, 75, 453, 0") ➞ 0

multiply\_nums("10, -2") ➞ -20

Ans.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def multiply\_nums(my\_string):

mult = 1

mylist = []

for i in my\_string.split(sep = ", "):

mylist.append(int(i))

for j in mylist:

mult \*= j

print(mult)

try:

multiply\_nums("2, 3")

multiply\_nums("1, 2, 3, 4")

multiply\_nums("54, 75, 453, 0")

multiply\_nums("10, -2")

lg.info("""Class multiply\_nums() has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

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.

Ans.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def square\_digits(my\_numb):

my\_numblist = list(str(my\_numb))

integer\_form = int

final\_result = ""

and\_in\_str = ""

for i in my\_numblist:

final\_result += str(int(i)\*\*2)

print(final\_result)

final\_int = int(final\_result)

return final\_int

try:

square\_digits(9119)

square\_digits(2483)

square\_digits(3212)

lg.info("""Class square\_digits() has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

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]

Ans.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def setify(my\_list):

answer = list(set(my\_list))

print(answer)

return answer

try:

setify([1, 3, 3, 5, 5])

setify([4, 4, 4, 4])

setify([5, 7, 8, 9, 10, 15])

setify([3, 3, 3, 2, 1])

lg.info("""Class setify() has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

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.

Ans.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def mean(numb):

total = int(len(str(numb)))

strnumb = str(numb)

sum = 0

for i in strnumb:

sum+= int(i)

avg = sum/total

print(int(avg))

try:

mean(42)

mean(12345)

mean(666)

lg.info("""Class mean() has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass