# Evidence 1

def ValidateHex(HexNumber):

if HexNumber == "": # empty string

return False # invalid

else:

for digit in HexNumber:

if (digit not in "0123456789ABCDEF"):

return False # not valid anymore

return True # only occurs if all the digits of the number are hexadecimal digits

def HexToDenDigit(HexDigit):

ValidCharacters = "0123456789ABCDEF"

for i in range(len(ValidCharacters)):

if HexDigit == ValidCharacters[i]:

return i # i (index) represents the value of the hexadecimal digit

return -1 # just in case...

def HexToDen(HexNumber):

HexNumber = HexNumber[::-1] # flip the digits to make them easier to process

result = 0 # initialise result

for i in range(len(HexNumber)): # for every index corresponding to a digit

digit = HexNumber[i] # represents the actual hexadecimal digit

value = HexToDenDigit(digit)

result += (value \* (16 \*\* i))

return result

def TestHexToDen():

number = input("Enter a hexadecimal number: ")

if (ValidateHex(number)):

print("The hexadecimal number {0} converted into decimal is {1}.".format(number, HexToDen(number)))

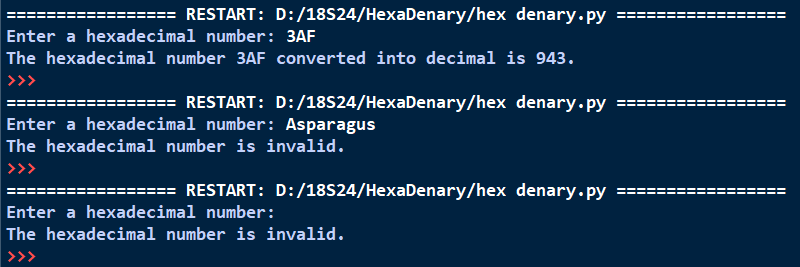
else:

print("The hexadecimal number is invalid.")

TestHexToDen()

# Evidence 2

|  |  |  |
| --- | --- | --- |
| Hexadecimal Number | Purpose of the test | Expected output |
| 3A | To test the function when the hexadecimal number is valid | 58 |
| Asparagus | To test the function when there are invalid digits | Invalid |
|  | To test the function when there is an empty string | Invalid |



# Evidence 3

def ValidateDen(DenNumber):

if DenNumber == "": # empty string

return False # invalid

else:

for digit in DenNumber:

if (digit not in "0123456789"):

return False # not valid anymore

return True

def DenToHexDigit(DenValue): # data type of DecValue: INTEGER

ValidCharacters = "0123456789ABCDEF"

return ValidCharacters[DenValue] # returns the hexadecimal digit representing DecValue

def DenToHex(DenNumber):

if not ValidateDen(DenNumber):

print("The denary number is invalid.") # the number is not valid

else:

number = int(DenNumber) # can be safely turned into an integer

result = "" # result is the hexadecimal number

while (number >= 1): # repeatedly do this procedure until number = 0

remainder = number % 16 # the remainder is the value of the hex digit, from the back

result += DenToHexDigit(remainder)

number //= 16 # get the next digit

result = result[::-1] # flip the expression to get the actual hexadecimal digit

print("The denary number {0} converted into hexadecimal is {1}.".format(DenNumber, result))

def TestDenToHex():

number = input("Enter a denary number: ")

DenToHex(number) # printing to output is handled within the function itself

TestDenToHex()

# Evidence 4

|  |  |  |
| --- | --- | --- |
| Denary Number | Purpose of the test | Expected output |
| 19 | To test the function when the denary number is valid | 13 |
| Banana | To test the function when there are invalid digits | Invalid |
|  | To test the function when there is an empty string | Invalid |

