Deborah Barndt

ITMD 513 Open Source Programming

Professor Dr. Sam

Hw3

2-9-19

'''

Deborah Barndt

2-6-19

CreditCardNumberValidation.py

hw3: Financial: Credit Card Number Validation

This program will ask the user to input their credit card number, then will

use Hans Luhn's algorithm to check if the card is valid or not valid. Once the

result is displayed, it will ask the user if they would like to check another

credit card number.

The algorithm is useful to determine whether the card number is entered correctly

or whether a credit card is scanned correctly by a scanner. The following steps are

used to check the credit card numbers: double every second digit from right to left.

If doubling of a digit results in a two-digit number, add up the two digits to get a

single-digit number, add all the single-digit numbers from the first step, add all

the digits in the odd places from right to left in the card number, sum the results

from the second and third steps, and if the result from the fourth step is divisible

by 10, the card number is valid; otherwise it is invalid.

Written by Deborah Barndt.

'''

# Main function to ask the user to enter a credit card number, then tell them if it

# is a valid or not valid credit card number and ask if they want to check another.

def main():

checkAgain = 'y'

while (checkAgain == 'y'):

number = int(input('Enter a credit card number between 13 and 16 digits: '))

if isValid(number):

print(number, 'is valid.')

# Ask user if they would like to check another credit card.

checkAgain = input('Would you like to check another credit card? (y/n) ')

else:

print(number, 'is not valid.')

# Ask user if they would like to check another credit card.

checkAgain = input('Would you like to check another credit card? (y/n) ')

# Return true if the card number is valid.

def isValid(number):

return getSize(number) >= 13 and getSize(number) <= 16 and \

(prefixMatched(number, 4) or prefixMatched(number, 5) or \

prefixMatched(number, 37) or prefixMatched(number, 6)) and \

(sumOfDoubleEvenPlace(number) + sumOfOddPlace(number)) % 10 == 0

# Get the result from Step 2.

def sumOfDoubleEvenPlace(number):

sum = 0

number = number // 10

while (number != 0):

sum += getDigit((number % 10) \* 2)

number = number // 100

return sum

# Return this number if it is a single digit, otherwise, return the sum

# of the two digits.

def getDigit(number):

return number % 10 + (number // 10)

# Return sum of odd place digits in number.

def sumOfOddPlace(number):

sum = 0

while (number != 0):

sum += number % 10

number = number // 100

return sum

# Return true if the digit d is a prefix for number.

def prefixMatched(number, d):

return getPrefix(number, getSize(d)) == d

# Return the number of digits in d.

def getSize(d):

length = 0

while (d != 0):

length += 1

d = d // 10

return length

# Return the first k number of digits from number. If the number of digits

# in number is less than k, return number.

def getPrefix(number, k):

total = number

for digit in range(getSize(number) - k):

total //= 10

return total

# Calling the main function to check the credit card number.

main()

Output Results:

