 **Northwestern Polytechnic University**

**Python Programming**

**Homework Assignment #2**

**Due day: 10/3/2021**

**Instruction:**

1. **Push the source code to Github or answer sheet in word file**
2. **Please follow the code style rule like programs on handout.**
3. **Overdue homework submission could not be accepted.**

**4. Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**

1. Create a function that takes an integer *m* greater than *1* and returns the largest integer smaller than *m* that evenly divides *m*.

***def lrgst\_factor(m):***

*"""Return the largest factor of m that is smaller than m*

*>>> lrgst\_factor (15) # factors are 1, 3, 5*

*5*

*>>> lrgst\_factor (80) # factors are 1, 2, 4, 5, 8, 10, 16, 20, 40*

*40*

*"""*

Answer : ***def lrgst\_factor(m):***

***num = 1***

***i = 2***

***while i<m:***

***if (m%i == 0):***

***num= i***

***i = i+1***

***return num***

1. Define a function which takes in a number *m* and determines whether the number is a perfect number. A perfect number is equal to the sum of all its factors. For instance, *6* is a perfect number since *6 = 1 + 2 + 3*.

***def pfct\_num(m):***

*"""*

*Returns True or False indicating whether* *"m" is a perfect*

*number. A number is a perfect number when the sum of all its*

*factors equal the number itself.*

*>>> pfct\_num (6) # 6 = 1 + 2 + 3*

*True*

*>>> pfct\_num (8) # 8 ≠1 + 2+ 4*

*False*

*>>> pfct\_num (28) # 28 =1 + 2 + 4 + 7 + 14*

*True*

*"""*

Answer : ***def pfct\_num(m):***

***total = 0***

***for x in range(1, m):***

***if m % x == 0:***

***total += x***

***return total == m***

1. Implement a function to check if the number of digits from two positive input parameters is the same or not.

***def same\_ord(a, b):***

*"""Return whether positive integers a and b have the same number of digits.*

*>>> same\_ord(50, 70) # 2 digits of a and b*

*True*

*>>> same\_ord(50, 100) # a has 2 digits; b has 3 digits*

*False*

*>>> same\_ord(1000, 100000) # a has 4 digits; b has 6 digits*

*False*

*"""*

Answer : ***def same\_ord(a,b):***

***Number= len(str(a))***

***Count = len(str(b))***

***while True:***

***if a<0:***

***return***

***elif b<0:***

***return***

***elif Count==Number:***

***return True***

***else:***

***return False***

1. Write a function that takes in a number and determines if the digits contain two adjacent *5s*.

***def two\_5(n):***

*"""Return true if n has two fives in a row.*

*>>> two\_5 (5)*

*False*

*>>> two\_5 (55)*

*True*

*>>> two\_5 (550055)*

*True*

*>>> two\_5 (12345)*

*False*

*>>> two\_5 (50505050)*

*False*

*"""*

Answer : ***def two\_5(n):***

***if n < 55:***

***return False***

***else:***

***return n % 100 == 55 or two\_5(n // 10)***

1. Design a function that returns the number of unique digits in a positive integer.

***def uniq\_digits(x):***

*"""Return the number of unique digits in positive integer x*

*>>> uniq\_digits (8675309) # All are unique*

*7*

*>>> uniq\_digits (1313131) # 1 and 3*

*2*

*>>> uniq\_digits (13173131) # 1, 3, and 7*

*3*

*>>> uniq\_digits (10000) # 0 and 1*

*2*

*>>> uniq\_digits (101) # 0 and 1*

*2*

*>>> uniq\_digits (10) # 0 and 1*

*2*

*"""*

Answer : ***def uniq\_digits(x):***

***unique = 0***

***if x = = 0:***

***print("no unique num!")***

***else:***

***print(len(set(str(x))))***

1. Write a *def* function *"amc"* with a positive integer *"n"* input parameter. It returns the smallest amicable number greater than *"n".* Two different numbers are both amicable if the sum of the proper divisors of each is equal to the other. Any number that's part of such a pair is an amicable number.

***Hint: You may want to create a separate function to sum proper divisors.***

***def amc(n):***

*"""*

*Return the smallest amicable number greater than positive integer n.*

*Every amicable number x has a buddy y different from x, such that*

*the sum of the proper divisors of x equals y, and the sum of the proper divisors of y equals x.*

*For example, 220 and 284 are both amicable because*

*1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 is 284, and*

*1 + 2 + 4 + 71 + 142 is 220*

*>>> amc(5)*

*220*

*>>> amc(220)*

*284*

*>>> amc(284)*

*1184*

*>>> r = amc(5000)*

*>>> r*

*5020*

*"""*

Answer : ***def amc(n):***

***i = 1***

***while 1>0:***

***sum1 = sod(n+i)***

***if sum1 != n+i:***

***sum2 = sod(sum1)***

***if sum2 == n+i :***

***print ( n+i )***

***break***

***i += 1***

***def sod(n):***

***rea = 1***

***per = n***

***i = 2***

***while i < n+1 :***

***sum = 1***

***if n%i == 0:***

***count = 1***

***while n%i == 0 :***

***sum +=   i\*\*count***

***n = n/i***

***count += 1***

***rea \*= sum***

***i += 1***

***return rea-per***

***amc(5)***