**WEEK-09**

1).Complete function to implement coin change making problem i.e. finding the minimum

number of coins of certain denominations that add up to given amount of money.

The only available coins are of values 1, 2, 3, 4

**Input Format:**

Integer input from stdin.

**Output Format:**

return the minimum number of coins required to meet the given target.

**Example Input**:

16

**Output:**

4

**Explanation:**

We need only 4 coins of value 4 each

**Example Input:**

25

**Output:**

7

**Explanation:**

We need 6 coins of 4 value, and 1 coin of 1 value

**PROGRAM:**

def coinChange(n):

dp = [float('inf')] \* (n + 1)

dp[0] = 0

coins = [1, 2, 3, 4]

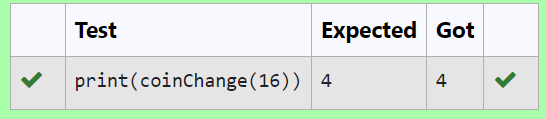
for coin in coins:

for i in range(coin, n + 1):

dp[i] = min(dp[i], dp[i - coin] + 1)

return dp[n]

**OUTPUT:**



2.Write a code to check whether product of digits at even places is divisible by sum of digits

at odd place of a positive integer.

**Input Format:**

Take an input integer from stdin.

**Output Format**:

Print TRUE or FALSE.

**Example Input:**

1256

**Output**:

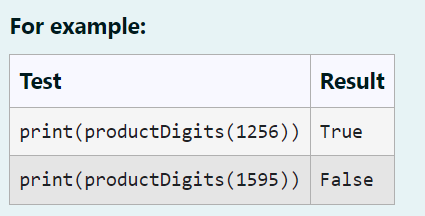
TRUE

**Example Input:**

1595

**Output:**

FALSE



**PROGRAM:**

def productDigits(number):

number\_str = str(number)

product\_even = 1

sum\_odd = 0

for i, digit\_char in enumerate(number\_str):

digit = int(digit\_char)

if (i+1)%2 == 0:

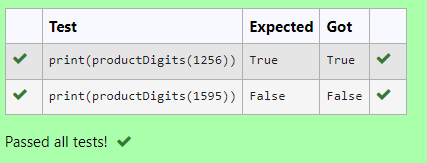
product\_even \*= digit

else:

sum\_odd += digit

return product\_even % sum\_odd == 0

**OUTPUT**:



3.A number is considered to be ugly if its only prime factors are 2, 3 or 5.

[1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, …] is the sequence of ugly numbers.

Task:

complete the function which takes a number n as input and checks if it's an ugly number.

return ugly if it is ugly, else return not ugly

Hint:

An ugly number U can be expressed as: U = 2^a \* 3^b \* 5^c, where a, b and c are nonnegative integers.

**For example:**

| **Test** | **Result** |
| --- | --- |
| print(checkUgly(6)) | ugly |
| print(checkUgly(21)) | not ugly |

**PROGRAM:**

def checkUgly(n):

if n <= 0:

return "not ugly"

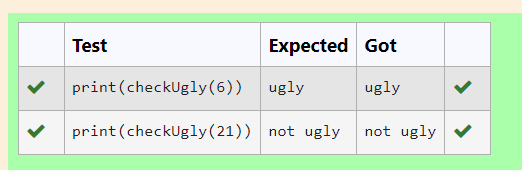
for p in [2, 3, 5]:

while n % p == 0:

n //= p

return "ugly" if n == 1 else "not ugly"

**OUTPUT:**



4. An abundant number is a number for which the sum of its proper divisors is greater than

the number itself. Proper divisors of the number are those that are strictly lesser than the number.

**Input Format:**

Take input an integer from stdin

**Output Format:**

Return Yes if given number is Abundant. Otherwise, print No

**Example input:**

12

**Output:**

Yes

**Explanation**

The proper divisors of 12 are: 1, 2, 3, 4, 6, whose sum is 1 + 2 + 3 + 4 + 6 = 16. Since sum of

proper divisors is greater than the given number, 12 is an abundant number.

**Example input:**

13

**Output:**

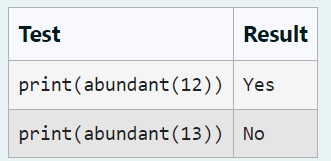
No

**Explanation**

The proper divisors of 13 is: 1, whose sum is 1. Since sum of proper divisors is not greater

than the given number, 13 is not an abundant number.

**For example:**

****

**PROGRAM:**

def abundant(number):

divisor\_sum=sum([divisor for divisor in range(1,number) if number % divisor==0])

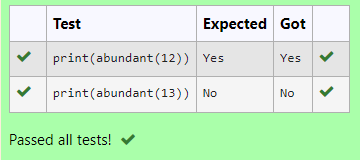
if divisor\_sum>number:

return "Yes"

else:

return "No"

**OUTPUT:**



5. An e-commerce company plans to give their customers a special discount for Christmas.

They are planning to offer a flat discount. The discount value is calculated as the sum of all

the prime digits in the total bill amount.

Write an algorithm to find the discount value for the given total bill amount.

Constraints

1 <= orderValue< 10e100000

Input

The input consists of an integer orderValue, representing the total bill amount.

Output

Print an integer representing the discount value for the given total bill amount.

Example Input

578

Output

12

**For example:**

| **Test** | **Result** |
| --- | --- |
| print(christmasDiscount(578)) | 12 |

**PROGRAM:**

def is\_prime(n):

if n <= 1:

return False

if n <= 3:

return True

if n % 2 == 0 or n % 3 == 0:

return False

i = 5

while i \* i <= n:

if n % i == 0 or n % (i + 2) == 0:

return False

i += 6

return True

def christmasDiscount(orderValue):

discount = 0

for digit in str(orderValue):

if is\_prime(int(digit)):

discount += int(digit)

return discount

try:

orderValue = int(input())

print(christmasDiscount(orderValue),end="")

except EOFError:

print(end="")

except ValueError:

print(end="")

**OUTPUT:**

