9- Functions

Example 1:

Input: s1 = "this apple is sweet", s2 = "this apple is sour"

Output: ["sweet","sour"]

Example 2:

Input: s1 = "apple apple", s2 = "banana"

Output: ["banana"]

 Constraints:

1 <= s1.length, s2.length <= 200

s1 and s2 consist of lowercase English letters and spaces.

s1 and s2 do not have leading or trailing spaces.

All the words in s1 and s2 are separated by a single space.

Note:

Use dictionary to solve the problem

For example:

Input	Result
this apple is sweet this apple is sour	sweet sour

Coin change

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

```
def coinChange(n):
    dp = [float('inf')] * (n + 1)
    dp[0] = 0
    coins = [1, 2, 3, 4]

    for i in range(1, n + 1):
        for coin in coins:
            if i - coin >= 0:
                dp[i] = min(dp[i], dp[i - coin] + 1)

    return dp[n]
```

For example:

Test	Result
print(coinChange(16))	4

Difference sum

Given a number with maximum of 100 digits as input, find the difference between the sum of odd and even position digits.

Input Format:

Take a number in the form of String from stdin.

Output Format:

Print the difference between sum of even and odd digits

Example input:

1453

Output:

1

Explanation:

Here, sum of even digits is 4 + 3 = 7

sum of odd digits is 1 + 5 = 6.

Difference is 1.

Note that we are always taking absolute difference

Program:

```
def differenceSum(n):
    num_str = str(n)

    sum_even = 0
    sum_odd = 0

    for i in range(len(num_str)):
        digit = int(num_str[i])
        if i % 2 == 0:
```

```
        sum_even += digit
    else:
        sum_odd += digit

  return abs(sum_even - sum_odd)
```

For example:

Test	Result
print(differenceSum(1453))	1

Automorphic number or not

An automorphic number is a number whose square ends with the number itself.

For example, 5 is an automorphic number because 5*5 =25. The last digit is 5 which same as the given number.

If the number is not valid, it should display "Invalid input".

If it is an automorphic number display "Automorphic" else display "Not Automorphic".

Input Format:

Take a Integer from Stdin

Output Format: Print Automorphic if given number is Automorphic number,otherwise Not Automorphic

Example input: 5

Output: Automorphic

Example input: 25

Output: Automorphic

 Example input: 7

Output: Not Automorphic

```python
def automorphic(n):
    if n <= 0:
        return "Invalid input"


    square = n * n


    num_last_digit = n % 10
```

```
square_last_digit = square % 10


if num_last_digit == square_last_digit:

    return "Automorphic"

else:

    return "Not Automorphic"
```

For example:

Test	Result
`print(automorphic(5))`	`Automorphic`

Ugly number

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.

Program:

```
def checkUgly(n):

    if n <= 0:

        return "not ugly"

    while n % 2 == 0:

        n //= 2

    while n % 3 == 0:

        n //= 3

    while n % 5 == 0:

        n //= 5

    if n == 1:

        return "ugly"
```

else:

   return "not ugly"

For example:

Test	Result
print(checkUgly(6))	ugly
print(checkUgly(21))	not ugly

Christmas Discount

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

Program:

```
def christmasDiscount(n):

  def is_prime(num):

    if num < 2:

      return False

    for i in range(2, int(num ** 0.5) + 1):

      if num % i == 0:
```

```
            return False

        return True


    discount = 0


    for digit in str(n):

        digit_int = int(digit)

        if is_prime(digit_int):

            discount += digit_int


    return discount
```

For example:

Test	Result
`print(christmasDiscount(578))`	12