# 06 - Strings in Python

**Ex. No. : 6.1 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Count Chars

Write a python program to count all letters, digits, and special symbols respectively from a given string

For example:

Input Result rec@123

3

3

1

PROGRAM:

a=input() c,d,s=0,0,0

for i in range(len(a)): if(a[i].isalpha()):

c+=1

elif(a[i].isdigit()): d+=1

else:

s+=1

print(c,d,s,sep="\n")



**Ex. No. : 6.2 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Decompress the String

Assume that the given string has enough memory. Don't use any extra space(IN- PLACE)

Sample Input 1 a2b4c6

Sample Output 1 aabbbbcccccc

PROGRAM:

import re a=input()

all=re.findall('\d+',a) all\_w=re.findall('[a-z]',a) b=''

for i,j in zip(all,all\_w): b+=int(i)\*j

print(b)



**Ex. No. : 6.3 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## First N Common Chars

Two string values S1, S2 are passed as the input. The program must print first N characters present in S1 which are also present in S2.

Input Format:

The first line contains S1. The second line contains S2. The third line contains N.

Output Format:

The first line contains the N characters present in S1 which are also present in S2. Boundary Conditions:

2 <= N <= 10

2 <= Length of S1, S2 <= 1000 Example Input/Output 1:

Input:

abcbde cdefghbb 3

Output:

bcd Note:

b occurs twice in common but must be printed only once.

PROGRAM:

a=input() b=input() C=''

d=int(input())

for i in range(len(a)): if(len(C)-d==0):

break else:

if(a[i]in b):

if(a[i] not in C): C+=a[i]

print (C)



**Ex. No. : 6.4 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Remove Characters

Given two Strings s1 and s2, remove all the characters from s1 which is present in s2.

Constraints

1<= string length <= 200

Sample Input 1 experience

enc

Sample Output 1 xpri

PROGRAM:

def remove\_chars(s1, s2):

return ''.join([char for char in s1 if char not in s2]) s1=input()

s2=input()

result = remove\_chars(s1, s2) print(result)



**Ex. No. : 6.5 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Remove Palindrome Words

String should contain only the words are not palindrome.

Sample Input 1

Malayalam is my mother tongue

Sample Output 1

is my mother tongue

For example:



PROGRAM:

a=[]

a=input() b=a. split() for i in b:

k=i.lower() if k!=k[::-1]:

print(k,end=' ')



**Ex. No. : 6.6 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Return Second World in Uppercase

Write a program that takes as input a string (sentence), and returns its second word in uppercase.

For example:

If input is “Wipro Technologies Bangalore” the function should return “TECHNOLOGIES”

If input is “Hello World” the function should return “WORLD” If input is “Hello” the program should return “LESS”

NOTE 1: If input is a sentence with less than 2 words, the program should return the

word “LESS”.

NOTE 2: The result should have no leading or trailing spaces. For example:

Input Result

Wipro Technologies Bangalore TECHNOLOGIES

Hello World WORLD

Hello LESS

PROGRAM:

f=input() s=f.split() if len(s)>1: c=s[1]

print(c.upper()) else:

print("LESS")



**Ex. No. : 6.7 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Revers String

Reverse a string without affecting special characters. Given a string S, containing special characters and all the alphabets, reverse the string without affecting the positions of the special characters.

Input:

A&B Output:

B&A

Explanation: As we ignore '&' and

As we ignore '&' and then reverse, so answer is "B&A".

For example:

Input Result A&x#

x&A#

PROGRAM:

def reverse\_string(s): s = list(s)

l, r = 0, len(s) - 1

while l < r:

if not s[l].isalpha(): l += 1

elif not s[r].isalpha(): r -= 1

else:

s[l], s[r] = s[r], s[l] l += 1

r -= 1

return ''.join(s) # Test Cases

print(reverse\_string(input())) # Output: "B&A"



**Ex. No. : 6.8 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## String characters balance Test

Write a program to check if two strings are balanced. For example, strings s1 and s2 are balanced if all the characters in the s1 are present in s2. The character’s position doesn’t matter. If balanced display as "true" ,otherwise "false".

For example:

Input Result Yn

PYnative True

PROGRAM:

a=input() b=input() if a in b:

print("True") else:

print("False")



**Ex. No. : 6.9 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Unique Names

In this exercise, you will create a program that reads words from the user until the user enters a blank line. After the user enters a blank line your program should display each word entered by the user exactly once. The words should be displayed in the same order that they were first entered. For example, if the user enters:

**Input:**

first second first third second

then your program should display:

**Output:**

first second third

PROGRAM: a,c=[],[]

for i in range(0,5):

b=input() a.append(b)

for i in range(len(a)): if(a[i] not in c):

c.append(a[i])

print(a[i])



**Ex. No. : 6.10 Date: 13/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Username Domain Extension

Given a string S which is of the format [USERNAME@DOMAIN.EXTENSION,](mailto:USERNAME@DOMAIN.EXTENSION) the program must print the EXTENSION, DOMAIN, USERNAME in the reverse order.

**Input Format**:

The first line contains S.

**Output Format**:

The first line contains EXTENSION. The second line contains DOMAIN. The third line contains USERNAME.

**Boundary Condition:**

1 <= Length of S <= 100 Example Input/Output 1: **Input**:

[vijayakumar.r@rajalakshmi.edu.in](mailto:vijayakumar.r@rajalakshmi.edu.in)

**Output**:

edu.in rajalakshmi vijayakumar.r

PROGRAM:

a = input()

ext = a.split('@')[0]

dom = a.split('@')[1].split('.')[0] userno = a.find('.')

user = a[userno+1:]

print(user) print(dom, end='\n') print(ext,end='\n')



# 07 - Functions

**Ex. No. : 7.1 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Abundant Number

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:

Test Result

print(abundant(12)) Yes

print(abundant(13)) No

PROGRAM:

def abundant(n): l,s=[],0

for i in range(1,int(n//2)+1): if(n%i==0):

l.append(i) for i in l:

s+=i if(s>n):

return("Yes") else:

return("No")



**Ex. No. : 7.2 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## 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

For example:

Test Result

print(automorphic(5)) Automorphic

PROGRAM:

def automorphic(n): a=str(n\*n) if(int(a[-1])==n):

return("Automorphic") else:

return("Not Automorphic")



**Ex. No. : 7.3 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## Check Product of Digits

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

For example:

|  |  |
| --- | --- |
| **Test** | **Result** |
| print(productDigits(1256)) | True |
| print(productDigits(1595)) | False |

def productDigits(n): PROGRAM:

a=str(n) s,p=0,1

for i in range(0,len(a),2): s+=int(a[i])

for i in range(1,len(a),2): p\*=int(a[i])

if(p%s==0): return("True")

else:

return("False") OUTPUT:

**Ex. No. : 7.4 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## 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 python code 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 christmasDiscount(n): res=0

while n!=0: rem=n%10 flag=0

for i in range(1,rem+1):

if rem%i==0: flag+=1

if flag==2: res=res+rem

n=n//10 return res

OUTPUT:



**Ex. No. : 7.5 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## 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

**PROGRAM:**

**def coinChange(amount):**

**# Available coin denominations coins = [1, 2, 3, 4]**

**# Initialize a list to store the minimum number of coins for each amount from 0 to the target amount**

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

**dp[0] = 0 # Base case: 0 coins needed to make amount 0**

**# Iterate through all amounts from 1 to the target amount for i in range(1, amount + 1):**

**# Iterate through all available coin denominations for coin in coins:**

**# If the current coin denomination is less than or equal to the current amount**

**if coin <= i:**

**# Update dp[i] to be the minimum between its current value and dp[i - coin] + 1**

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

**# The result is stored at dp[amount] return dp[amount]**

**amount = int(input()) print(coinChange(amount))**



**Ex. No. : 7.6 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## 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): a=[]**

**b=[]**

**k=str(n)**

**for i in range(len(k)): if int(i)%2==0:**

**a.append(int(k[i])) else:**

**b.append(int(k[i])) s=sum(b)**

**r=sum(a) j=s-r return j**

**Ex. No. : 7.7 Date: 20/4/2024**

**Register No: 231501008 Name: AFRAH M**

## 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.

**For example:**

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

**PROGRAM:**

**def checkUgly(n):**

**for i in range(n):**

**for j in range(n):**

**for k in range(n):**

**if(n==(2\*\*i)+(3\*\*j)+(5\*\*k)): return("ugly")**

**return("not ugly")**

