**Instacks Practice Codes**

**Hello Instacks**

Write a Python Program to print Hello Instacks

print(“Hello Instacks”)

**Output:**

Hello Instacks

**Print Statements**

Write a Python program to print the following statement

print(“Welcome to Instacks Python programming”)

print(“It is a great language to learn and so simple”)

print(“Let us have FUN and learn”)

**Output:**

Welcome to Instacks Python programming

It is a great language to learn and so simple

Let us have FUN and learn

**Read and Print Input Statement**

Write a Python program to read the input at run time and print the input statements.

s = input()

print(s)

**Output:**

Let us have FUN and learn

Let us have FUN and learn

**First and Last name of user**

Write a python program to read the first name and last name of the user and print them in output

first = input()

last = input()

print(first,last)

**Output:**

Anand

K

Anand K

**Read and Print 5 inputs with comma separated**

Write a Python program to read 5 input numbers at run time and print them with comma separated.

first = int(input())

second = int(input())

third = int(input())

fourth = int(input())

fifth = int(input())

print(first, second, third, fourth, fifth, sep = ‘,’)

**Output:**

20

25

8

5

9

20,25,8,5,9

**Average of numbers**

Write a Python program to Calculate the Average of Numbers in a Given List.

**Note:** Print till 2 digits after decimal point

n = int(input())

a = []

for i in range(0,n):

elem = int(input())

a.append(elem)

avg = sum(a)/n

print(round(avg,2))

**Output:**

2

10

15

12.50

**Integer Representation in Binary, Octal, and Hexadecimal Values**

Write a program to read an integer number from the user and print its equivalent Binary, Octal, and Hexadecimal values.

n = int(input())

print(bin(n))

print(oct(n))

print(hex(n))

**Output:**

100

0b1100100

0o144

0x64

**Character to ASCII**

Write a Python program to print the ASCII value of a given character.

a = input()

print(ord(a)

**Output:**

A

65

**ASCII to Character**

Write a Python program to print the ASCII Character for the given integer value.

a = int(input())

print(chr(a))

**Output:**

35

#

**Circle Computations**

Write a Python program which accepts the radius of a circle from the user and compute the diameter, circumference, and area of a Circle.

pi = 3.14

r = float(input())

diameter = 2 \* r

circumference = 2 \* pi \* r

area = pi \* r\*\*2

print(“%.2f” %diameter)

print(“%.2f” %circumference)

print(“%.2f” %area)

**Output:**

1.1

2.20

6.91

3.80

**File Extension**

Write a Python program to accept a filename from the user and print the extension of that file.

filename =input()

f\_extns = filename.split(“.”)

print(f\_extns[-1])

**Output:**

myText.txt

txt

**Volume of the Sphere**

Write a Python program to calculate the volume of a sphere by getting the value from the user.

pi =3.14

r = float(input())

v = 4.0/3.0 \* pi \* r\*\*3

print(“%.2f” %v)

**Output:**

6

904.32

**Swapping**

Write a Python program to read two values from the user and interchange the values and display the values.

a = input()

b = input()

a,b =b,a

print(a)

print(b)

**Output:**

10

20

20

10

**Simple Interest**

Write a Python program to calculate the simple Interest by taking the values dynamically from the user.

p = int(input())

t = int(input())

r = int(input())

I = (p \* t \* r) / 100

print(int(i))

**Output:**

25000

2

5

**Height Conversion**

Write a Python program to dynamically read the height of a person in centimetres and then converts the height to feet and inches and print them.

cm = input()

inches = 0.394 \* cm

feet = 0.328 \* cm

print(“%.2f” %(inches))

print(“%.2f” %(feet))

**Output:**

153

60.28

50.18

**N+NN+NNN**

Write a Python program to read a number n from the user and compute the series: n + nn + nnn

n = int(input())

temp = str(n)

s1 = temp + temp

s2 = temp + temp + temp

res = n + int(s1) + int(s2)

print(res)

**Output;**

5

615

**Quotient and Remainder**

Write a Python program read 2 Integer Numbers and print their Quotient and Remainder.

a = int(input())

b = int(input())

quotient = a // b

remainder = a % b

print(quotient)

print(remainder)

**Output:**

15

7

2

1

**Area of Triangle**

Write a Python program to calculate the Area of a Triangle.

a = int(input())

b = int(input())

c = int(input())

s = (a + b + c) / 2

area = (s \* (s – a) \* (s - b) \* (s – c)) \*\* 0.5

print(“%.2f” %area)

**Output:**

4

5

6

9.92

**Sales Commission**

A Computer manufacturing company has the following monthly compensation policy to the sales-person:

Minimum base salary Rs.15000

Bonus for every computer sold Rs.2000

Commission on total monthly sale 2%

Since the prices of the computers are changing, the sales price of each computer is fixed at the beginning of every month.

Write a Python program to compute a sales-person’s Gross salary.

**Note:** Output to be rounded off to 2 digits.

# Gross Salary = Base Salary + (quantity \* bonus rate) + (quantity \* price) \* Commission rate

base\_salary = 15000

bonus\_rate = 2000

commission = 0.02

quantity =int(input())

price = float(input())

bonus = bonus\_rate \* quantity \* price

gross\_salary = base\_salary + bonus + commission

print(“%.2f” %bonus)

print(“%.2f” %commission)

print(“%.2f” %gross\_salary)

**Output:**

20

45500.25

1820010000.00

0.02

1820025000.02

**Literacy Calculations**

In a town, the percentage of men is 52. The percentage of total literacy is 48. If total percentage of literate men is 35 of the population, write a program to find the total number of literate men and women if the population of the town is 80,000.

tot\_pop = 80000

literacy = 48

men\_literacy = 35

tot\_lit = (80000 \* 48) // 100

men = (38400 \* 35) // 100

women = tot\_lit – men

print(tot\_pop)

print(tot\_lit)

print(men)

print(women)

**Output:**

80000

38400

13440

24960

**Kilometre Conversions**

The distance between two cities (in km.) is input through the keyboard. Write a Python program to convert and print this distance in meters, feet, inches, and centimetres.

km = float(input())

m = km \* 1000

cm = m \* 100

inches = cm / 2.54

feet = inches / 12

print(“%.2f” %m)

print(“%.2f” %cm)

print(“%.2f” %feet)

print(“%.2f” %inches)

**Output:**

25

25000.00

2500000.00

82021.00

984251.97

**Calculate Gross**

Ramesh’s basic salary is input through the keyboard. His dearness allowance is 40% of basic salary, and house rent allowance is 20% of basic salary.

Write a Python program to calculate his gross salary.

basic = float(input())

dearness = basic / 2.5

house\_rent = basic / 5

gross\_pay = basic + dearness + house\_rent

print(“%.2f” %basic)

print(“%.2f” %dearness)

print(“%.2f” %house\_rent)

print(“%.2f” %gross\_pay)

(or)

basic = float(input())

dearness = 0.4 \* basic

house\_rent = 0.2 \* basic

gross\_pay = basic + dearness + house\_rent

print(“%.2f” %basic)

print(“%.2f” %dearness)

print(“%.2f” %house\_rent)

print(“%.2f” %gross\_pay)

**Output:**

10000

10000.00

4000.00

2000.00

16000.00

**Calculate Currency**

A Cashier has currency notes of denominations 100’s,50’s,20’s,10’s,5’s,2’s and 1’s. If the amount to be withdrawn is input through the keyboard in hundreds, find the total number of currency notes of each denomination the cashier will have to give to the withdrawer.

amount = int(input())

total = amount // 100

print(total)

amount = amount – (total \* 100)

total = amount // 50

print(total)

amount = amount – (total \* 50)

total = amount // 20

print(total)

amount = amount – (total \* 20)

total = amount // 10

print(total)

amount = amount – (total \* 10)

total = amount // 5

print(total)

amount = amount – (total \* 5)

total = amount // 2

print(total)

amount = amount – (total \* 2)

total = amount // 1

print(total)

**Output:**

565

565

65

15

15

5

0

0

**Second Conversion**

Write a Python program to read the total number of seconds from the user and convert it into the following format – Hours:Minutes:Seconds

sec = int(input())

hours = (sec // 3600)

minutes = (sec – (3600 \* hours)) // 60

seconds = (sec – (3600 \* hours) – (minutes \* 60))

print(f”{hours}:{minutes}:{seconds}”)

**Output:**

25300

7:1:40

**Year Conversions**

Write a Python program to convert specified days into years, weeks and days.

**Notes:** Ignore leap year

days = int(input())

years = days // 365

weeks = (days % 365) // 7

days = days – ((years \* 365) + (weeks \* 7))

print(f”{years}”)

print(f”{weeks}”)

print(f”{days}”)

**Output:**

1329

3

33

3

**Winning Percentage**

Write a Python program that calculates and displays the winning percentage of a team by taking the number of wins, losses and matches drawn by the team.

won = int(input())

lost = int(input())

draw = int(input())

played = won + lost + draw

percentage = (won / played) \* 100

print(round(percentage, 2))

**Output:**

98

55

5

62.03

**Laptop Purchase**

Consider that you have purchased a Laptop – price get it from the user upon which a sales tax of 6% is applied. Write a Python program that calculates and displays the total purchase price of the Laptop.

price = float(input())

sales\_tax = 6

total\_price = ((price \* sales\_tax) / 100) + price

print(“%.2f” %total\_price)

**Output:**

45800

48548.00

**Generate Number**

Write a program to input a single digit (n) and print a 3 digit number created as e.g., if you input 7, then it should print 789.

n = input()

num = n + str(int(n) + 1) + str(int(n) + 2) # we have converted the number to int type

to add 1&2 and then converted back to str

type for concatenation

print(num)

**Output:**

7

789

**Compound Interest**

Write a Python program to calculate the Compound Interest.

p = float(input())

r = float(input())

t = float(input())

ci = p \* (pow((1 + r / 100), t))

print(ci)

**Output:**

1000

5

2

1102.5

**Rectangle Perimeter and area**

Write a Python program that will obtain the length and width of a rectangle from the user and compute its area and perimeter.

l = int(input())

w = int(input())

a = l \* w

p = 2 \* (l + w)

print(p)

print(a)

**Output:**

20

40

120

800

**Control Statements**

**Return the Product**

Accept two integer values from the user and return their product. If the product is greater than 1000, then return their sum.

Def multiplication\_or\_sum(num1, num2):

product = num1 \* num2

if product <= 1000:

return product

else:

return num1 + num2

number1 = int(input())

number2 = int(input())

result = multiplication\_or\_sum(number1, number2)

print(result)

**Output:**

5

8

40

**Generate a Range**

Given a range of numbers. Iterate from 1st number to the end number and print the sum of the current number and previous number.

Def sumNum(num):

previousNum = 0

for i in range(num):

sum = previousNum + i

print(i, previousNum,sum)

previousNum = i

end = int(input())

sumNum(end)

**Output:**

10

0 0 0

1 0 1

2 1 3

3 2 5

4 3 7

5 4 9

6 5 11

7 6 13

8 7 15

9 8 17

**Even Characters**

Accept string from the user and display only those characters which are present at an even index

def printEvenIndexChar(str):

for i in range(0, len(str), 2):

print(i, str[i])

inputStr = input()

printEvenIndexChar(inputStr)

**Output:**

python

0 p

2 t

4 o

**Divisible by 5**

Given a list of numbers, iterate it and print only those numbers which are divisible of 5

def findDivisible(numberList):

for num in nimberList:

if (num % 5 == 0):

print(num)

numList = eval(input())

findDivisible(numList)

**Output:**

10, 20, 23, 46, 55

10

20

55

**Natural Numbers**

Write a Python program to print first N natural numbers using while loop

n = int(input())

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

print(i, end = “ “)

**Output:**

10

1 2 3 4 5 6 7 8 9 10

**Sum of 1 to N**

Write a Python program to accept number from user and calculate the sum of all number from 1 to a given number.

n = int(input())

sum = 0

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

sum +=i

print(sum)

**Output:**

10

55

**Multiplication Table**

Write a Python program to print the Multiplication table for the given number

a = int(input())

n = int(input())

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

mul = a \* i

print(f”{a} \* {i} = {mul}”)

**Output:**

3

10

3 \* 1 = 3

3 \* 2 = 6

3 \* 3 = 9

3 \* 4 = 12

3 \* 5 = 15

3 \* 6 = 18

3 \* 7 = 21

3 \* 8 = 24

3 \* 9 = 27

3 \* 10 = 30

**Divisible by 5 and greater than 150**

Write a Python Program to iterate over a given list, and display only the numbers that are divisible by five, and if you find a number greater than 150, stop the loop iteration

list1 = eval(input())

for item in list1:

if (item > 150):

break

if (item % 5 == 0):

print(item, end = “ “)

**Output:**

[12, 15, 32, 42, 55, 75, 122, 132, 150, 180, 200]

15 55 75 150

**Count the Digits**

Write a Python program to get an /integer number from the user and count the number of digits within the number and print the result.

num = int(input())

temp = num

count = 0

while temp != 0:

temp //= 10

count += 1

print(count)

**Output:**

75869

5

**Prime Numbers within the Range**

Write a Python program to the Prime Numbers between a given range of numbers (both inclusive).

**Note:** A Prime number is whole number that cannot be made by multiplying other whole numbers.

**Example:**

* 6 is not a Prime number because it can be made by 2 \* 3 = 6
* 37 is a Prime number because no other whole numbers multiply together to make it

start = int(input())

end = int(input())

for num in range(start, end + 1):

**# all prime numbers are greater than 1**

**# if number is less than or equal to 1, it is not prime**

if num > 1:

for I in range(2, num):

**#check for factors**

if (num % i) == 0:

**# not a prime number so break inner loop and**

**# look for next number**

break

else:

print(num, end = “ “)

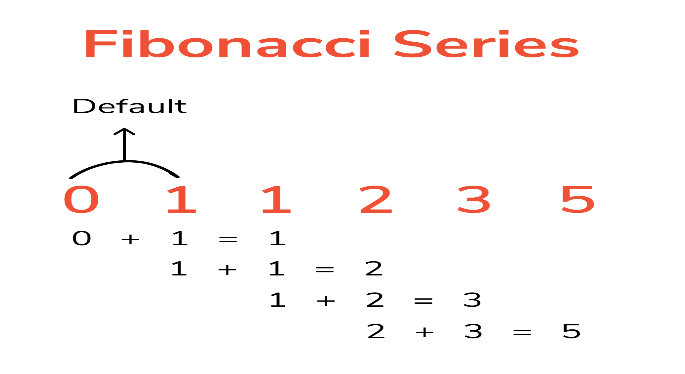
**Output:**

5

10

5 7

**Fibonacci Series**

Write a program to display the Fibonacci Series up to the specified term. 

terms = int(input())

# first two terms

num1, num2 = 0, 1

count = 0

while count < terms:

print(num1, end = “ “)

temp = num1 + num2

# update values

num1 = num2

num2 = temp

count += 1

**Output:**

10

0 1 1 2 3 5 8 13 21 34

**Factorial of a number**

Write a Python program to find the factorial of a given number.

And if the given number is a Negative Number print -1 as the result.

The factorial (Symbol : !) means to multiply all whole numbers from the chosen number down to 1.

**Example:** Calculate the factorial of 5

5! = 5 \* 4 \* 3 \* 2 \* 1 = 120

num =int(input())

factorial = 1

if num < 0:

print(“-1”)

elif num == 0:

print(“1”)

else:

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

factorial = factorial \* i

print(factorial)

**Output:**

5

120

**Reverse a Number**

Write a Python program to read an integer number and print the reverse of the number.

num = int(input())

reverse\_number = 0

while num > 0:

remainder = num % 10

reverse\_number = (reverse\_number \* 10) + remainder

num = num // 10

print(reverse\_number)

**Output:**

321

123

**Number Palindrome**

Write a Python program to read an integer number dynamically from the user and check whether the given number is a Palindrome or not. If the number is a Palindrome then print 1 as the result otherwise 0.

Number Palindrome means the reverse of the number should be same as the given number itself.

n = int(input())

oNum = n

rNum = 0

while(n > 0):

remainder = n % 10

rNum = (rNum \* 10) + remainder

n = n // 10

if(oNum == rNum):

print(“1”)

else:

print(“0”)

**Output:**

121

1

**List Odd Index**

Write a Python program to read a list from the user and print only the values that are present in the odd index position of the list.

my\_list = eval(input())

for i in my\_list[1::2]:

print(i, end = “ “)

**Output:**

[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

20 40 60 80 100

**Cubes**

Write a Python program to read an integer value from the user and display the cube of the number up to the given value.

n = int(input())

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

cube = i \*\* 3

print(f”{i} {cube}”)

**Output:**

6

1 1

2 8

3 27

4 64

5 125

6 216

**Tax Calculation**

Calculation income tax for the given income by adhering to the below rules

Taxable Income Rate (%)

First $10,000 0

Second $10,000 10

The remaining 20

income = int(input())

taxPayable = 0

if income <= 10000:

taxPayable = 0

elif income <= 20000:

taxPayable = (income – 10000) \* 10 / 100

else:

# first 10000

taxPayable = 0

# next 10000

taxPayable = 10000 \* 10 / 100

# remaining

taxPayable += (income – 20000) \* 20 / 100

print(taxPayable)

**Output:**

28000

2600.0

**Exponent Function**

Write a function called exponent (base, exp) that returns an int value of base raises to the power of exp.

**Note:** Here exp is a non-negative integer, and the base is an integer.

def exponent(base, exp):

num = exp

result = 1

while num > 0:

result = result \* base

num = num -1

print(result)

# Drive Code

b = int(input())

e = int(input())

exponent(b, e)

**Output:**

5

5

3125

**Sum of String Numbers**

Write a Python program to find the sum of digits in the inputted String.

string1 = input()

sum\_digit = 0

for x in string1:

if x.isdigit() == True:

z = int(x)

sum\_digit = sum\_digit + z

print(sum\_digit)

**Output:**

python123welcome4

10

**Character Counts**

Write a Python program to count Uppercase, Lowercase, special character and numeric values in a given string.

str = input()

upper\_ctr, lower\_ctr, number\_ctr, special\_ctr = 0, 0, 0, 0

for i in range(len(str)):

if str[i] >= “A” and str[i] <= “Z”:

upper\_ctr += 1

elif str[i] >= “a” and str[i] <= “z”:

lower\_ctr += 1

elif str[i] >= “0” and str[i] <= “9”:

number\_ctr += 1

else:

special\_ctr += 1

print(str)

print(upper\_ctr)

print(lower\_ctr)

print(number\_ctr)

print(special\_ctr)

**Output:**

PythonIsCool@123

PythonIsCool@123

3

9

3

1

**Armstrong Number**

Write a Python program to check whether the given number is an Armstrong number or not.

If the given number is an Armstrong number then print 1 otherwise print 0

**Armstrong number:**

Abcd… = pow(a, n) + pow(b, n) + pow(c, n) + pow(d, n) + …

Where n represents the number of digits.

num = int(input())

# Change num variables to string, and calculated the number of digits

order = len(str(num))

sum = 0

# find the sum of the cube of each digit

temp = num

while temp > 0:

digit = temp % 10

sum += digit \*\* order

temp //= 10

# display the result

if num == sum:

print(“1”)

else:

print(“0”)

**Output:**

153

1

**Prime Number**

Write a Python program to check whether the given number is Prime Number or not. If the given number is a Prime Number print 1 otherwise print 0 as the result.

A **prime number** is a natural number greater than 1 that has no positive divisors other than 1 and itself.

num = int(input())  
# If given number is greater than 1

if num > 1:

# Iterate from 2 to n / 2

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

# If num is divisible by any number between

# 2 and n / 2, it is not prime

if (num % i) == 0:

print(“0”)

break

else:

print(“1”)

else:

print(“0”)

**Output:**

11

1

**Check Fibonacci**

Write a Python program to check whether the given number is a Fibonacci Number or not.

**Fibonacci numbers** are 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 141, ...

(Fibonacci series is the sum of the previous two terms)

import math

# function that returns true if x is perfect square

def isPerfectSquare(x):

s = int(math.sqrt(x))

return s \* s == x

# Returns true if n is a Fibonacci Number, else false

# n is Fibonacci if 5 \* n \* n – 4 or both is a perfect square

def isFibonacci(n):

return isPerfectSquare(5 \* n \* n + 4) or isPerfectSquare(5 \* n \* n -4)

# A utility function to test above functions

i = int(input())

if (isFibonacci(i) == True):

print(“1”)

else:

print(“0”)

**Output:**

8

1

**Sum of Square**

Write a Python program to find the Sum of first n natural numbers.

Square: 12 + 22 + 32 + … + N2.

def squaresum(n):

sum = 0

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

sum = sum + (i \* i)

return sum

n = int(input())

print(squaresum(n))

**Output:**

4

30

**Sum of Cubes**

Write a Python program to find the Cube sum of first n natural numbers.

Cube: 13 + 23 + 33 + 43 + …+ n3 till nth term.

def cubesum(n):

sum = 0

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

sum = sum + (i \*\* 3)

return sum

n = int(input())

print(cubesum(n))

**Output:**

5

225

**Date Checker**

Write a Python program that in a date and checks if it a valid date and prints the incremented date if it is.

* If the given month is invalid (Lesser than 1 or Greater than 12) – print 0
* If the given date is invalid (There are some month that have only 30 days, some months with 31 days and Feb has either 28 or 29) – print -1

date = input()

dd, mm, yy = date.split(‘/’)

dd = int(dd)

mm = int(mm)

yy = int(yy)

if(mm == 1 or mm == 3 or mm == 5 or mm == 7 or mm == 8 or mm == 10 or mm == 12):

max1 = 31

elif(mm == 4 or mm ==6 or mm == 9 or mm == 11):

max1 = 30

elif(yy % 4 == 0 and yy % 100 != 0 or yy % 400 == 0):

max1 = 29

else:

max1 = 28

if(mm < 1 or mm > 12):

print(“0”)

elif(dd < 1 or dd > max1):

dd = 1

mm = mm + 1

print(dd, mm, yy)

elif(dd == 31 and mm == 12):

dd = 1

mm = 1

yy = yy + 1

print(dd, mm, yy)

else:

dd = dd + 1

print(dd, mm, yy)

**Output:**

5/7/2016

6 7 2016

**Leap Year**

Write a Python program to check whether a given year is a leap year or not

def checkYear(year):

if(year % 4 == 0 and year % 100 != 0 or year % 400 == 0):

print(“1”)

else:

print(“0”)

year = int(input())

checkYear(year)

**Output:**

2016

1

**Perfect Number**

Write a Python program to check whether a given number is a Perfect Number or not.

**Perfect Number:** It is a positive integer that is equal to the sum of its positive divisors, excluding the number itself

**For example:**

6 is the first perfect number

Proper divisors of 6 are 1, 2, 3

Sum of its proper divisors = 1 + 2 + 3 = 6

Hence 6 is a perfect number.

n = int(input())

sum = 0

for i in range(1, n):

if (n % i == 0):

sum += i

if (sum == n):

print(“1”)

else:

print(“0”)

**Output:**

6

1

**Strong Number**

Write a Python program to check whether a given number is a Strong Number or not.

**Strong Number** is a special number whose sum of factorial of digits is equal to the original number.

sum = 0

num = int(input())

temp = num

while (num):

i =1

f = 1

r = num % 10

while (i <= r):

f \*= i

i += 1

sum += f

num //= 10

if(sum == temp):

print(“1”)

else:

print(“0”)

**Output:**

145

1

**Strong Number in 1 to n**

Write a Python program to print the Strong numbers between 1 to N

# Store the factorial of all the digits from [0, 9]

factorial = [1, 1, 2, 6, 24, 120, 720, 5040, 40320, 362880]

# Function to check whether the number is Strong or not

def isStrong(N):

num = str(N)

sum = 0

for i in range(len(num)):

sum += factorial[ord(num[i]) – ord(‘0’)]

if sum == N:

return True

else:

return False

# Function to print all Strong Numbers between the given range

def printStrongNumbers(N):

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

if(isStrong(i)):

print(I, end = “ ”)

N = int(input())

printStrongNumbers(N)

**Output:**

100

1 2

**GCD**

Write a Python program to read two numbers from the user and find the GCD (or) HCF of the two numbers and print as a result.

**HCF** (Highest Common Factor) is also known as **GCD** (Greatest Common Divisor) or **GCF**(Greatest Common Factor).

When we find all the factors of two or more numbers, and some factors are the same (“common”), then the largest of those common factors is the Greatest Common Factor.

A diagram of a common factor

Description automatically generated

def compute\_hcf(x, y):

# choose the smaller number

if x > y:

smaller = y

else:

smaller = x

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

if ((x % i == 0) and (y % i == 0)):

hcf = i

return hcf

n1 = int(input())

n2 = int(input())

print(compute\_hcf(n1,n2))

**Output:**

12

30

6

**Amicable numbers**

Write a Python program to read two numbers from the user and find whether they are amicable numbers or not

**Amicable numbers** or friendly numbers are a pair of numbers whose sum of proper divisors equals the other.

x = int(input())

y = int(input())

sum1 = 0

sum2 = 0

for i in range(1, x):

if x % i == 0:

sum1 += i

for j in range(1, y):

if y % j == 0:

sum2 += j

if (sum1 == y and sum2 ==x):

print(“1”)

else:

print(“0”)

**Output:**

220

284

1

**Magic Number -1**

Write a Python program to read an integer number and check whether the given number is a Magic number or not. If it is a Magic number print1 otherwise print 0.

**What are Magic Numbers?**

A number is said to be a magic number, if the sum of its digits are calculated till a single digit recursively by adding the sum of the digits after every addition, If the single digits comes out to be 1, then the number is a magic number.

n = int(input())

sum = 0

temp = n

while (n > 0 or sum > 0):

if (n == 0):

n = sum

sum = 0

sum += n % 10

n //= 10

if (sum == 1):

print(“1”)

else:

print(“0”)

**Output:**

55

1

**Automorphic number**

Write a Python program to read a number from the user and check whether it is a Automorphic number or not. If it is a automorphic number then print 1 otherwise print 0.

**Automorphic number** is a number whose square of number end digit with the same digits as the original number.

num = int(input())

# calculating the number of digits

num\_of\_digits = len(str(num))

# computing the square of a number

square = num \*\* 2

# obtaining the last digits

last\_digits = square % pow(10, num\_of\_digits)

# comparing the digits of number with input

if last\_digits == num:

print(“1”)

else:

print(“0”)

**Output:**

5

1

**Harshad number**

Write a Python program to read a number from the user and check whether if it is a Harshad number or not. If the number is a Harshad number, then print 1 otherwise print 0.

**Harshad number** is a number that is divisible by the sum of its digits.

num = int(input())

rem = sum = 0

# Make a copy of num and store it in variable n

n = num

# Calculates sum of digits

while (num > 0):

rem = num % 10

sum += rem

num //= 10

# Check whether the number is divisible by the sum of digits

if (n % sum == 0):

print(“1”)

else:

print(“0”)

**Output:**

21

1

**Magic Number - 2**

Write a Python program to read an Integer number and check whether the given number is a Magic number or not. If it is a Magic number print 1 otherwise print 0.

**Write are Magic Number?**

A magic number is that number whose sum of the digits is when multiplied by the reverse of the same sum results back the original number.

A math equations on a white background

Description automatically generated

num = int(input())

temp = num

sumOfDigits = 0

rev = 0

# Calculating Sum of digits

while (temp > 0):

# Extract digits and add them

sumOfDigits += temp % 10

temp //= 10

temp = sumOfDigits

# Compute reverse of Sum of Digits

while (temp > 0):

rev = rev \* 10 + temp % 10

temp //= 10

if (rev \* sumOfDigits == num):

print(“1”)

else:

print(“0”)

**Output:**

81

1

**Abundant number**

Write a Python program to read an integer from the user and check whether the given number is an Abundant number or not. If it is a Abundant number print 1 otherwise print 0.

A Number is an **Abundant number** if it’s lesser than (smaller) the sum of all its proper divisors. And the difference between these two values is called the abundance.

import math

# Function to calculate sum of divisors

def getSum(n):

sum = 0

# Note that this loop runs till square root of n

i = 1

while i <= (math.sqrt(n)):

if n % i == 0:

# If divisors are equal, take only one of them

if n / i == i:

sum += i

else:

# Otherwise take both

sum += i

sum += (n / i)

i += 1

# Calculate sum of all proper divisors only

sum -= n

return sum

# Function to check Abundant Number

def checkAbundant(n):

# Return true if sum of divisors is greater than n

if (getSum(n) > n):

return 1

else:

return 0

# Driver program to test above function

n = int(input())

if(checkAbundant(n) == 1):

print(“1”)

else:

print(“0”)

**Output:**

12

1

**Factors of a Number**

Write a Python program to input a number from user and find all factors of the given number.

**Factor** of any number is a whole number which exactly divides the number into a whole number without leaving any remainder.

num = int(input())

# Iterate from 1 to num

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

# If num is exactly divisible by i

# Then i is a factor of num

if(num % i == 0):

print(i, end = ‘ ‘)

**Output**

12

1 2 3 4 6 12

**Swap First and Last**

Write a Python program to read an integer number from the user and to swap the first and last digit of the number.

If the Number is a Single Digits Number then print -1 as the result.

num = int(input())

rev = 0

noOfDigit = 0

temp = num

while temp > 0:

temp = int(temp/10)

noOfDigit = noOfDigit + 1

if noOfDigit < 2:

print(“-1”)

elif noOfDigit == 2:

temp = num

while temp > 0:

rem = temp % 10

rev = (rev \* 10) + rem

temp = int(temp/10)

print(rev)

else:

temp = num

while temp > 0:

rem = temp % 10

rev = (rev \* 10) + rem

temp = int(temp/10)

revNum =rev

rev = 0

temp = num

noOfDigitTemp = noOfDigit

while temp > 0:

remTemp = revNum % 10

if noOfDigitTemp == noOfDigit:

rem = temp % 10

rev = (rev \* 10) + rem

elif noOfDigitTemp == 1”

rem = temp % 10

rev = (rev \* 10) + rem

else:

rev = (rev \* 10) + remTemp

temp = int(temp/10)

revNum = int(revNum/10)

noOfDigitTemp = noOfDigitTemp – 1

print(rev)

**Output:**

12345

12345

**Leap Year in a Range**

Write a Python program to read the starting and ending year values and print the Leap years present between these two given years.

# get the start year from user

startYear = int(input())

# get the end year from user

endYear = int(input())

# loop through between the start and end year

for i in range(startYear,endYear + 1, 1):

# check if the (i) year is a Leap year if yes print

if ((0 == i % 4) and (0 != i % 100) or (0 == i % 400)):

print(i)

**Output:**

2000

2005

2000

2004

**Dictionary**

**Dict-1**

Write a Python program to create a dictionary with Every character in the string as a key and its frequency as the value for the key

S = input()

D = {}

for c in S:

if c not in D:

D[c] = 1

else:

D[c] = D[c] + 1

Print(D)

**Output:**

Programming

{‘P’: 1, ‘r’: 2, ‘o’: 1, ’g’: 2, ‘a’: 1, ‘m’: 2, ‘i’: 1, ‘n’: 1}

**Dict-2**

Write a Python program to merge 3 different dictionaries and create a new dictionary and print it.

D1 = eval(input())

D2 = eval(input())

D3 = eval(input())

New\_D = {}

for d in (D1, D2, D3):

New\_D.update(d)

Print(New\_D)

**Output:**

{1: 10, 2: 20}

{5: 50, 6: 60, 7: 70}

{‘A’: 1, ‘B’: 2}

{1: 10, 2: 20, 5: 50, 6: 60, 7: 70, ‘A’: 1, ‘B’: 2}

**Dict-3**

Write a Python program to add a new key-value pair into the dictionary after checking if the key already exists or not.

If the key is not present -get the value for that key, insert the key-value into the dictionary and print the mpdified dictionary as a result.

If the key already exists, then print the appropriate message – Key already Present

D = eval(input())

k = eval(input())

v = eval(input())

if k in D:

print(‘Key already Present’)

else:

D.update({k: v})

print(D)

**Output:**

{‘A’: 1,’B’: 2, ‘C’: 3, ’D’: 4}

‘E’

5

**Dict-4**

Write a Python Program to get a dictionary value from the user and count the number of keys that are having the same value and print the count.

D = eval(input())

V = eval(input())

res = 0

for key in D:

if D[key] == V:

res = res + 1

print(res)

**Output:**

{‘A’: 1, ‘B’: 2, ’C’:3, ‘b’: 2, ‘bee’: 2}

2

3

**Dict-5**

Write a Python program to find the Sum of all items in a dictionary and print the result.

D = eval(input())

sum = 0

for i in D:

sum += D[i]

print(sum)

**Output:**

{‘A’: 100, ’B’: 200, ‘C’: 300, ‘D’: 400}

1000

**Dict-6**

Write a Python program to create a dictionary with key starting from lower\_limit to upper\_limit and its value is the cube of that number

D = {}

l = int(input())

u = int(input())

for x in range(l, u + 1):

D[x] = x \*\* 3

print(d)

**Output:**

1

10

{1: 1, 2: 8, 3: 27, 4: 64, 5: 125, 6: 216, 7: 343, 8: 512, 9: 729, 10: 1000}

**Dict-7**

Write a Python program to sort a dictionary – Based on the Keys

D = eval(input())

for i in sorted (D.keys()):

print(f”{i} {D[i]}”)

**Output:**

{2: 56, 1: 12, 5: 122, 4: 24, 6: 18, 3: 323}

1 12

2 56

3 323

4 24

5 122

6 18

**Dict-8**

Write a Python program to a sort a dictionary – Based on the Values

D = eval(input())

L = sorted(D.items(), key = lambda kv:(kv[1], kv[0]))

D = dict(L)

for x, y in D.items():

print(f”{x} {y}”)

**Output:**

{2: 56, 1: 12, 5: 122, 4: 24, 6: 18, 3: 323}

1 12

6 18

4 24

2 56

5 122

3 323

**Dict-9**

Write a Python program to combine two dictionary and create a new dictionary – key will be the common key between the two dictionary and its value is the sum of the 2 common key from both the dictionaries

D1 = eval(input())

D2 = eval(input())

D3 = {}

for x in D1:

for y in D2:

if x == y:

D3[x] =D1[x] + D2[y]

print(D3)

**Output:**

{‘A’: 25, ‘B’: 87, ‘C’: 125, ‘D’: 87}

{‘B’: 425, ‘D’: 872, ‘J’: 125, ‘E’: 7}

{‘B’: 512, ‘D’: 959}

**Dict-10**

Write a Python program to read two lists from the user and create a dictionary by mapping the two lists and print the resultant dictionary.

keys = eval(input())

values = eval(input())

# empty dictionary

res\_dict = dict()

for i in range(len(keys)):

res\_dict.update({keys[i]: values[i]})

print(res\_dict)

**Output:**

[‘red’, ‘green’, ‘blue’]

[‘#FF0000’, ‘#008000’, ‘#0000FF’]

{‘red’: ‘#FF0000’, ‘green’: ‘#008000’, ‘blue’: ‘#0000FF’}

**Dict-11**

Write a Python program to check whether the dictionary is empty or not? If empty print 0 otherwise print the length of the dictionary.

D = eval(input())

n = len(D)

if n > 0:

print(n)

else:

print(“0”)

**Output:**

{}

0

**Dict-12**

Write a Python program to find the depth of a dictionary

def dict\_depth(dic, level = 1):

str\_dic = str(dic)

counter = 0

for i in str\_dic:

if i == “{“:

counter +=1

return(counter)

dic = eval(input())

print(dict\_ddepth(dic))

**Output:**

{1: ‘Java’, 2: {3: {4: {}}}}

4

**Dict-13**

Write a Python program to print all the unique values present in the dictionary.

dict1 = eval(input())

res = list(sorted({ele for val in dict1.values() for ele in val}))

print(res)

**Output:**

{‘red’: ‘#FF0000’, ‘green’: ‘#008000’, ‘blue’: ‘#0000FF’}

[‘#’, ‘0’, ‘8’, ’F’]

**Dict-14**

As a trainer i teach so many different languages to the students, today i thought of taking a poll.

I wanted to know which language my students liked the most.

The choices were –

C

Java

Python

Which ever language gets the most number of votes is declared as the winner

And if 2 languages share the same number of votes, we give whichever language’s name is shorter as the winner

Now write a program to help me validate the votes and declared as the winner.

from collections import Counter

def winner(input):

temp = Counter(input)

votes = dict(temp)

print(votes)

d = {}

for value in votes.values():

d[value] = []

for (key, value) in votes.items():

d[value].append(key)

maxVote = sorted(d.keys(), reverse = True)[0]

if len(d[maxVote]) > 1:

print(sorted(d[maxVote])[0])

else:

print(d[maxVote][0])

input = eval(input())

winner(input)

**Output:**

[“python”, ”java”, ”c”, ”java”, ”python”, ”c”, ”java”, ”java”, ”c”, ”python”, ”c”, ”c”, ”java”, ”java”, ”python”, ”java”, ”python”, ”python”, ”java”, ”c”, ”python”, ”java”]

{‘python’: 7, ‘java’: 9, ‘c’: 6}

java

**Dict-15**

Write a Python program to a create a new dictionary by extracting the mentioned keys and create a new dictionary and print it as the result.

sample\_dict = eval(input())

keys = eval(input())

res = dict()

for k in keys:

res.update({k: sample\_dict[k]})

print(res)

**Output:**

{“name”: “Kelly”, “age”: 25, ”salary”: 8000, “city”: “New York”}

[“name”, “salary”]

{‘name’: ‘Kelly’, ‘salary’: 8000}

**Dict-16**

Write a Python program to read a dictionary and a list.

Delete the list of keys from the dictionary and then print the modified dictionary as the result.

# Using the pop() method and loop

Sample\_dict = eval(input())

Keys = eval(input())

for k in keys:

sample\_dict.pop(k)

print(sample\_dict)

**Output:**

{“name”: “Kelly”, ”age”: 25, “salary”: 8000, “city”: “New York”}

[“name”, “salary”]

{‘city’: ‘New York’, ‘age’: 25}

**Dict-17**

Write a Python program to read a dictionary with values list and extract key whose value has most unique values.

test\_dict = eval(input())

max\_val = 0

max\_key = None

for sub in test\_dict:

# test for length using len()

# converted to set for duplicates removal

if len(set(test\_dict[sub])) > max\_val:

max\_val = len(set(test\_dict[sub]))

max\_key = sub

print(max\_key)

**Output:**

{“ABC”: [5, 7, 9, 4, 0], ”is”: [6, 7, 4, 3, 3], “Best”: [9, 9, 6, 5, 5]}

ABC

**Dict-18**

Write a Python program to remove all duplicate values across all the dictionary values lists.

form collections import Counter

test\_dict = eval(input())

# Remove duplicate values across Dictionary Values

# Using Counter() + list comprehension

cnt = Counter()

for idx in test\_dict.values():

cnt.update(idx)

res = {idx: [key for key in j if cnt[key] == 1] for idx, j in test\_dict.items()}

print(res)

**Output:**

{‘Manjeet’: [1], ‘Akash’: [1, 8, 9]}

{‘Manjeet’: [], ‘Akash’: [8, 9]}

**Dict-19**

Write a program to check whether a given key exists in a dictionary or not. If present then return True and print the value of the Key.

d = eval(input())

key = eval(input())

if key in d.keys():

print(“True”)

print(d[key])

else:

print(“False”)

**Output:**

{“S1”: [60, 85, 75, 90, 75], “S2”: [85, 85, 95, 75, 85], ”S3”: [85, 45, 90, 78, 98]}

“S3”

True

[85, 45, 90, 78, 98]

**Dict-20**

Write a program to get the maximum and minimum value of dictionary.

a\_dictionary = eval(input())

# get key with max value

max\_key = max(a\_dictionary, key = a\_dictionary.get)

min\_key = min(a\_dictionary, key = a\_dictionary.get)

print(max\_key)

print(min\_key)

**Output:**

{“a”: 1, “b”: 2, “c”: 3}

c

a

**Functions**

**Stuttering Function**

Write a function that stutters a word as if someone is struggling to read it. The first two letters are repeated twice with an ellipsis … and space after each, and then word is pronounced with a question mark?

def stutter(word):

return “{0}… {0}… {1}?”.format(word[:2], word)

s = input()

print(stutter(s))

**Output:**

incredible

in,,, in… incredible?

**Curzon Numbers**

Write a Python program to establish if a given integer num is a Curzon number. If 1 plus 2 elevated to num is exactly divisible by 1 plus 2 multiplied by num, then num is a Curzon number.

def is\_curzon(n):

return not (2 \*\* n + 1) % (2 \* n + 1)

num = int(input())

print(is\_curzon(num))

**Output:**

5

True

**Basic Calculator**

Create a function that two numbers and a mathematical operator + - / \* and will perform a calculation with the given numbers.

def calculator(num, num2, operator):

if operator == “+”:

return num1 + num2

elif operator == “-“:

return num1 – num2

elif operator == “\*”:

return num1 \* num2

elif operator == “/”:

if num2 == 0:

return “Can’t divide by 0!”

else:

return num1 // num2

n1 = int(input())

n2 = int(input())

o = input()

print(calculator(n1, n2, o))

**Output:**

2

2

+

4

**Return the Factorial**

Create a function that takes an integer and returns the factorial of that integer.

def factorial(num):

return 1 if num < 2 else num \* factorial(num -1)

n1 = int(input())

print(factorial(n1))

**Output:**

10

3628800

**Convert a Number…**

Create a function that returns a base -2 (binary) representation of a base -10 (decimal) string number.

def binary(d):

return bin(d)[2:]

n = int(input())

print(binary(n))

**Output:**

5

101

**Count Ones in Binary…**

Count the amount of ones in the binary representation of an integer.

def count\_ones(num):

return bin(num).count(‘1’)

n = int(input())

print(count\_ones(n))

**Output:**

12

2

**X’s and O’s**

Create a function that takes a string, checks if it has the same number of “x”s and “o”s and returns either True or False.

def XO(text):

x = 0

o = 0

for ch in text:

if ch in ‘xX’:

x += 1

elif ch in ‘oO’:

o += 1

if x == o:

return True

else:

return False

s = input()

print(XO(s))

**Output:**

ooxx

True

**Correct Inequality**

Create a function that returns True if a given inequality expression is correct and False otherwise.

def correct\_signs(txt):

return eval(txt)

s = input()

print(correct\_signs(s))

**Output:**

3 < 7 < 11

True

**ATM PIN Code Value…**

ATM machines allow 4 or 6 digits PIN codes and PIN codes cannot contain anything but exactly 4 digits or exactly 6 digits. You task is to create a function that takes a string and returns True if the PIN is valid and False if it’s not.

def is\_valid\_PIN(pin):

return len(pin) in [4, 6] and pin.isdigit()

s = input()

print(is\_valid\_PIN(s))

**Output:**

1234

True

**End Corona!**

Create a function that the number of daily average recovered cases recovers, daily average new\_cases, current active\_cases, and returns the number of days it will take to reach zero cases.

def end\_corona(recovers, new\_cases, active\_cases):

count = 0

while active\_cases > 0:

active\_cases -= recovers

active\_cases += new\_cases

count += 1

return count

n1 = int(input())

n2 = int(input())

n3 = int(input())

print(end\_corona(n1, n2, n3))

**Output:**

4000

2000

77000

39

**Luke, I am Your**

Luke Skywalker has family and friends. Help him remind them who is who. Given a string with a name, return the relation of that person to Luke.

|  |  |
| --- | --- |
| **Person** | **Relation** |
| Darth Vader | father |
| Leia | sister |
| Han | brother in law |
| R2D2 | droid |

def relation\_to\_luke(name):

solution = {‘Lela’: ‘sister’,

‘Darth Vader’: ‘father’,

‘Han’: ‘brother in law’,

‘R2D2’: ‘droid’}

Return ‘Luke, I am your {}.’.format(solutions[name])

s = input()

print(relation\_to\_luke(s))

**Output:**

Darth Vader

Luke, I am your father.

**Error Messages**

Create a function that takes a number as an argument and returns the appropriate error message.

The input error will be 1 to 5:

1 >> “Check the fan”

2 >> “Emergency stop”

3 >> “Pump Error”

4 >> “c”

5 >> “Temperature Sensor Error”

def error(n):

return {

1: ‘Check the fan: e1’,

2: ‘Emergency stop: e2’,

3: ‘Pump Error: e3’,

4: ‘c: e4”,

5: ‘Temperature Sensor Error: e5”,

}.get(n, 101)

e = int(input())

print(error(e))

**Output:**

1

Check the fan: e1

**Default Mood**

Create a function that in a current mood and return a sentence in the following format: “Today, I am feeling {mood}”. However, if no argument is passed, return “Today, I am feeling neutral”.

def mood\_today(\*mood):

if len(mood) != 0:

return “Today, I am feeling {}”.format(\*mood)

else:

return “Today, I am feeling neutral”

s = input()

print(mood\_today(s))

**Output:**

happy

Today, I am feeling happy

**Flip the Boolean**

Create a function that reverses a Boolean value and returns the string “Boolean expected” if another variable type is given.

def reverse(arg):

if str(arg) == “True”:

return False

elif str(arg) == ”False”:

return True

return “boolean expected”

s = eval(input())

print(reverse(s))

**Output:**

True

False

**H4ck3r Sp34k**

Create a function that takes a string as an argument and returns a coded (h4ck3r 5p34k) version of the string.

#’a’s with 4, ‘e’s with 3, ‘i’s with 1, ‘o’s with 0, and ‘s’s with 5

def hacker\_speak(txt):

return txt.replace(‘a’, ’4’).replace(‘e’, ‘3’).replace(‘i’, ‘1’).repalce(‘o’, ‘0’).replace(‘s’, ’5’)

s = input()

print(hacker\_speak(s))

**Output:**

javascript is cool

j4v45cr1pt 15 c00l

**Sum of all numbers**

Lists can be mixed with various types. Your task for this challenge is to sum all the number elements in the given list. Create a function that takes a list and returns the sum of all numbers in the list.

def numbers\_sum(lst):

return sum(i for i in lst if type(i) == int)

s = eval(input())

print(numbers\_sum(s))

**Output:**

[1, 2, “13”, “4”, “645”]

3

**Even Number Generator**

Create a function that finds all even numbers from 1 to the given number.

def find\_even\_nums(num):

return [x for x in range(2, num + 1, 2)]

s = int(input())

print(find\_even\_nums(s))

**Output:**

8

[2, 4, 6, 8]

**Filter Strings**

Create a function that takes a list of strings and integers and filters out the list so that it returns a list of integers only.

def filter\_list(l):

new = []

for el in l:

if type(el) == type(1):

new.append(el)

return new

s = eval(input())

print(filter\_list)

**Output:**

[1, 2, 3, “a”, “b”, 4]

[1, 2, 3, 4]

**Stand in Line**

Write a function that takes a list and a number as arguments. Add the number to the end of the list, then remove the first element of the list.

The function should then return the updated list.

def next\_in\_line(lst, num):P

if lst == []:

return “No list has been selected”

else:

lst.pop(0)

lst.append(num)

return lst

s1 = eval(input())

s2 = eval(input())

print(next\_in\_line(s1, s2))

**Output:**

[5, 6, 7, 8, 9]

1

[6, 7, 8, 9, 1]

**Play BlackJack**

In BlackJack, cards are counted with -1, 0, 1 values:

2, 3, 4, 5, 6 are counted as +1

7, 8, 9 are counted as 0

10, J, Q, K, A are counted as -1

Create a function that sums the number and returns it from the list of cards provided.

def count(deck):

d1 = {2:1, 3:1, 4:1, 5:1, 6:1, 7:0, 8:0, 9:0, 10:-1, ‘J’:-1, ‘Q’:-1, ‘K’:-1, ‘A’:-1}

r = []

for x in deck:

if x in d1:

r.append(d1[x])

return sum(r)

s = eval(input())

print(count(s))

**Output:**

[5, 9, 10, 3, “J”, “A”, 4, 8, 5]

1

**Western Showdown**

Wild Roger is participating in a Western Showdown, meaning he has to draw (pull out and shoot) his gun faster than his opponent in a gun standoff. Given two strings, p1 and p2, return which person drew their gun the fastest. If both are drawn at the same time, return “tie”.

def showdown(p1, p2):

if p1.index(‘B’) == p2. Index(‘B’):

return “tie”

elif p1.index(‘B’) > p2. Index(‘B’):

return “p2”

else:

return “p1”

s1 = input()

s2 = input()

print(showdown(s1, s2))

**Output:**

“Bang!”

“Bang!”

p1

**Correct Inequality**

Create a function that returns True if a given inequality expression is correct and False otherwise.

def correct\_signs(txt):

return eval(txt)

s = input()

print(correct\_signs(s))

**Output:**

3 < 7 < 11

True

**Find the Discount**

Create a function that takes two arguments: the original price and the discount percentage as integers and returns the final price after the discount.

def dis(price, discount):

return price \* (100 – discount) \* 0.01

p = int(input())

d = int(input())

print(dis(p, d))

**Output:**

100

75

25.0

**How Many Vowels?**

Create a function that takes a string and returns the number (count) of vowels contained within it.

def count\_vowels(txt):

return sum([1 for x in txt.lower() if x in ‘aeiou’])

s = input()

print(count\_vowels(s))

**Output:**

Prediction

4

**ASCII of Inverse Case**

Create a function that takes a single character as an argument and returns the char code of its lowercased / uppercased counterpart.

def counterpartCharCode(char):

return ord(char.swapcase())

s = input()

print(counterpartCharCode(s))

**Output:**

a

6

**Parity Check**

Create a function that takes a number as input and returns True if the sum of its digits has the same parity as the entire number. Otherwise, return False.

def parity\_analysis(num):

k = [int(no) for no in str(num)]

if num % 2 == 0:

return sum(k) % 2 == 0

else:

return sum(k) % 2 !=0

s = int(input())

print(parity\_analysis(s))

**Output**

243

True

**Sum of Evenly Divided**

Create a function that takes three arguments a, b, c and returns the sum of the numbers that are evenly divided by c from the range a, b inclusive.

def evenly\_divisible(a, b, c):

return sum(i for i in range(a, b + 1) if not i % c)

n1 = int(input())

n2 = int(input())

n3 = int(input())

print(evenly\_divisible(n1, n2, n3))

**Output:**

1

10

2

30

**Circle or Square**

Given the radius of a circle and the area of a square, return True if the circumference of the circle is greater than the square’s perimeter and False if the square’s perimeter is greater than the circumference of the circle.

def circle\_or\_square(r, s):

return 1.57 \* r > s \*\* 0.5

r = int(input())

s = int(input())

print(circle\_or\_square(r, s))

**Output:**

16

625

True

**HTML Tag Repetition**

Write a Python program to a read a String Input from the user as shown below and produce the given output as the result.

def secret(t):

a, b = t.split(‘\*’)

return ‘<%s></%s>’%(a,a)\*int(b)

s = input()

print(seceret(s))

**Output:**

div \* 2

<div><div></div></div>

**Calculating Damage**

Write a Python program to read 3 values from the user – damage, speed (attacks per second), mode and returns the amount of damage after a given time.

def damage(damage, speed, time):

if speed < 0 or damage < 0:

return “-1”

if time == “second”:

return int(damage \* speed)

if time == ”minute”:

return int(damage \* speed \* 60)

if time == “hour”:

return int(damage \* speed \* 3600)

d = eval(input())

s = eval(input())

t = input().lower()

print(damage(d, s, t))

**Output:**

40

5

second

200

**Strings**

**String Mid-1**

Given a string of odd length greater than or equal to 7, return a string made of the middle three chars of a given String. And if the string length is less than 7, return the original string value itself as the output.

s = input()

n =len(s)

if n >= 7:

mid = n // 2

ms = s[mid - 1 : mid + 2]

print(ms)

else:

print(s)

**Output**

Anushya

ush

**String Mid-2**

Given 2 strings, s1, and s2, create a new string by appending s2 in the middle of s1.

s1 = input()

s2 = input()

if len(s1) % 2 == 0:

mid = len(s1) // 2

ms = s1[:mid] + s2 + s1[mid:]

print(ms)

else:

mid = len(s1) // 2

ms = s1[:mid + 1] + s2 + s1[mid + 1:]

print(ms)

**Output:**

Python

Java

pytjavahon

**Mix String-1**

Given 2 strings, s1, s2 return a new string made of the first, middle and last char each input strings.

def mix\_string(s1, s2):

first\_char = s1[:1] + s2[:1]

middle\_char = s1[int(len(s1) / 2): int(len(s1) / 2) + 1] +s2[int(len(s2) / 2): int(len(s2) / 2) + 1]

last\_char =s1[len(s1) – 1] + s2[len(s2) -1]

res = first\_char + middle\_char + last\_char

print(res)

s1 = input()

s2 = input()

mix\_string(s1, s2)

**Output:**

America

Japan

AJrpan

**Mix String-2**

Write a python program to arrange the string characters such that all the lowercase letters should come first followed by all the uppercase letters in each string. If the string contains anything other than alphabets ignore those characters.

s = input()

lower = []

upper = []

for char in s:

if char.islower():

lower.append(char)

elif char.isupper():

upper.append(char)

ns = ‘ ’.join(lower + upper)

print(ns)

**Output:**

pYtHOn

ptnYHO

**Count Strings**

Write a Python program to count all the Lowercase, Uppercase, Digits and Special symbols from a given string.

def findCount(inputString):

ucount = 0

lcount = 0

digitCount = 0

symbolCount = 0

for char in inputString:

if char.islower():

lcount += 1

elif char.isupper():

ucount += 1

elif char.isnumeric():

digitCount += 1

else:

symbolCount += 1

print(lcount)

print(ucount)

print(digitCount)

print(symbolCount)

inputString = input()

findCount(inputString)

**Output:**

Anushya. [1083@gmail.com](mailto:1083@gmail.com)

1

14

4

3

**String Mix-3**

Write a Python program to take 2 strings as input and create a new string by mixing the 1st char of string one with the last character of string two and next the 2nd character of string one with the last before character of string two and so on. If any characters are left out, put them at the end of the newly created string.

def mixString(s1, s2):

s2 = s2[::-1]

lengthS1 = len(s1)

lengthS2 = len(s2)

length = lengthS1 if lengthS1 > lengthS2 else lengthS2

resultString = “”

for i in range(length):

if(i < lengthS1):

resultString = resultString + S1[i]

if(i < lengthS2):

resultString = resultString + s2[i]

print(resultString)

s1 = input()

s2 = input()

mixString(s1, s2)

**Output:**

Python

Java

PayvtahJon

**S1 Present in S2**

Write a Python program to read 2 string values from the user and check whether all the characters of String1 are present in String@.

**Note:** You should check for all characters in string1 present in string2 but no need to bother about the order in which they appear.

def string\_balance\_test(s1, s2):

flag = True

for char in s1:

if char in s2:

continue

else:

flag = False

return flag

s1 = input()

s2 = input()

flag = string\_balance\_test(s1, s2)

print(flag)

**Output:**

hut

hunter

True

**USA Occurrences**

Write a Python program to find all occurrences of the sub-string “USA” in given string ignoring the case if the sub-string is not present return the count as 0.

s = input()

sub = “USA”

t = s.lower()

count = t.count(sub.lower())

print(count)

**Output:**

Welcome to USA. usa awesome, isn’t it?

2

**Extract Numbers**

Write a Python program to read the string in the below mentioned format and print the sum and average of the digits that appear in the string, ignoring all other characters.

l = input().split()

sum = 0

l2 = []

n = 0

a = ‘ ’

for i in l:

for x in i:

if x.isdigit():

a = a + x

print(a)

sum = sum + int(a)

n = n + 1

a = ‘ ’

avg = sum / n

print(sum)

print(avg)

**Output:**

English = 98 Science = 80 Math = 100 History = 65

98

80

100

65

343

85.75

**Character Frequency**

Write a Python program to read a string as an input and count occurrences of all characters within a string

s = input()

countDict = dict()

for char in s:

count = s.count(char)

countDict[char] = count

print(countDict)

**Output:**

PythonPython

{‘P’:2, ‘y’:2, ‘t’:2, ‘h’:2, ‘o’:2, ‘n’:2}

**String Reverse**

Write a Python program to read a String value dynamically abd reverse the given string value.

s = input()

# Method 🡪 Using the Slicing

str1 = s[::-1]

print(str1)

**Output:**

Instacks

skcatsmI

**Last Occurrence**

Write a Python program to find the last position of a substring “Emma” in a given string.

And if the words “Emma” is not present within the string then return -1 as the result.

str1 = input()

index = str1.rfind(“Emma”)

print(index)

**Output:**

Emma is a data scientist who knows Python. Emma works at google.

43

**Split String-1**

Write a Python program to split a given string on hyphens into several substrings and display each sub-string.

str1 = input()

# split string

substrings = str1.split(“-“)

# Displaying each substring

for sub in substrings:

print(sub)

**Output:**

Emma-is-a-data-scientist

Emma

is

a

data

scientist

**Remove Empty String**

Write a Python program to remove empty strings from the list of strings.

str\_list = eval(input())

# use built-in function filter to filter empty value

new\_str\_list = list(filter(None, str\_list))

print(new\_str\_list)

**Output:**

[‘Emma’, ‘Jon’, ‘Kelly’, None, ‘Eric’, ‘’]

[‘Emma’, ‘Jon’, ‘Kelly’, ‘Eric’]

**Remove Special Character**

Write a Python program to Remove special symbols/ Punctuation from a given string.

import string

str1 =input()

# use translate function of a string

# and maketrans function of str class

new\_str = str1.translate(str.maketrans(‘’, ‘’, string.punctuation))

print(new\_str)

**Output:**

/\*Jon is @ developer & musician

Jon is developer

**Only Digits**

Write a Python program to remove all the characters other than the digits from the given string and print the numbers.

str1 = input()

# Retain Numbers in string

# Using list comprehension + join() + isdigit()

res = “”.join([item for item in str1 if str1 if item.isdigit()])

print(res)

**Output:**

I am 25 years and 10 months old

2510

**Alphanumberic String**

Write a Python program to find the words which contains both characters and digits in them from the given string and print it as result.

str1 = input()

# words with both alphabets and numbers

# isdigit() for numbers + isalpha() for alphabets

# use any() to check each character

res = []

temp = str1.split()

for item in temp:

if any(char.isalpha() for char in item) and any(char.isdigit() for char in item):

res.append(item)

# Displaying words with alphabets and numbers

For i in res:

print(i)

**Output:**

Emma25 is Data scientist50 and AI Expert

Emma25

scientist50

**Replace with #**

Write a Python program to remove from the given string and replace each punctuation with a # symbol.

**Examples of different punctuation marks include:**

Full stops (.),

Commas (,),

Question marks (?),

Exclamation marks (!),

Colons (:),

Semi-colons (;),

Apostrophes (‘),

Speech marks (“,”).

From string import punctuation

str1 = input()

# Replace punctuation with #

replace\_char = ‘#’

# Using string.punctuation to get the list of all punctuations

# use string function replace() to replace each punctuation with #

for char in punctuation:

str1 = str1.replace(char, replace\_char)

print(str1)

**Output:**

/\* Jon is @developer & musician

## Jon is # developer # musician

**Remove N Characters**

Given a string and an integer n, remove characters from string starting from zero up to n and return a new string.

**Note:** n must be less than the length of the String.

s = input()

n = int(input())

r = s[n:]

print(r)

**Output:**

python

3

Hon

**Reverse every second word**

Write a program to read a String value from the user and reverse every second word in the string.

str = input()

lst = str.split()

length = len(lst)

list\_2 = []

# Reverse the word present at the Odd index position

for i in range(0, length, 1):

if(i % 2 == 0):

list\_2.append(lst[i])

else:

list\_1.append(lst[i][::-1])

output = ‘’.join(list\_2)

print(output)

**Output:**

One Two Three Four Five

One owT Three ruoF Five

**String Palindrome**

Write a Python program to find whether the given string is a Palindrome or not.

def reverse(str1):

if(len(str1) == 0):

return str1

else:

return reverse(str1[1:]) + str1[0]

string = input()

str1 =reverse (string)

print(str1)

if(string == str1):

print(“1”)

else:

print(“0”)

**Output:**

madam

madam

1

**String Reverse-1**

Write a Python program to reverse the order of words in a string.

str = input()

# By default split() divides the string based on blank space

lst = str.split()

r = lst[::-1]

# printing the words in form of a string

output = ‘’. join(r)

print(output)

**Output:**

Python is a cool Language

Language cool a is Python

**String Reverse-2**

Write a Python program to reverse the internal content of each word

str = input()

lst = str.split()

# New List to hold the reversed words of the string

list\_2 = []

for word in lst:

list\_2.append(word[::-1])

output = ‘’.join(list\_2)

print(output)

**Output:**

HiMate Soft Solutions

etaMiH tfos snoituloS

**Alphabet Repeat**

Read every character present in the string, if it is an alphabet save the alphabet in a variable and if it is a digit, take the previous alphabet stored in the variable and repeat it the number of times the digit is.

Repeat this process till it reaches the end of the string.

# Enter your String alphabet followed by digit

string = input()

output\_Str = “”

for ch in string:

if ch.isalpha():

x = ch

else:

output\_Str = output\_Str + x \* int(ch)

print(output\_Str)

**Output:**

Anu5

uuuuu

**String Length**

Write a program to find the length of the String without using the built-in function

def string\_length(str1):

count = 0

for char in str1:

count += 1

return count

string = input()

print(string\_length(string))

**Output:**

Welcome to Python

17

**String Mix-3**

Write a Python program to create a new string made up of the first 2 and last 2 characters of a given string.

**Note:**

* If the length of the string is less than 2 return -1
* If the length of the string is exactly 2 characters, then the 1st 2 and the last 2 characters would be the same.

def New\_String(str):

if len(str) < 2:

newStr = “-1”

return newStr

else:

newStr = str[0:2] + str[-2:]

return newStr

s = input()

print(New\_String(s))

**Output:**

Welcome to Python

Weon

**String Mix-3**

Write a Python program to create a new single string from two given strings, by swapping the first two characters of each string.

Order while printing – 1st String followed by the 2nd string value without any space in between.

def StringMix(a, b):

new\_a = b[:2] + a[2:]

new\_b = a[:2] + b[2:]

return new\_a + ‘ ’ + new\_b

string1 = input()

string2 = input()

print(StringMix(string1, string2))

**Output:**

Sai

Anu

AniSau

**Remove a Character**

Write a program to remove an element from the specified index position and return the new string.

def remove\_index(str, i):

first\_part = str[:i]

last\_part = str[i + 1:]

return first\_part + last\_part

string = input()

index = int(input())

print(remove\_index(string, index))

**Output:**

Core Java

3

Cor Java

**String Mix-3**

Write a Python program to create a new string from a String where the first and last chars have been exchanged.

def Exchange(str1):

return str1[-1:] + str1[1:-1] +str1[:1]

string = input()

print(Exchange(string))

**Output:**

Sairam

mariaS

**Remove Odd**

Write a Python program to remove the elements in the odd index position and return the new String.

def oddIndex(Str):

result = “”

for i in range(len(str):

if i % 2 == 0:

result = result +str[i]

return result

string = input()

print(oddIndex(string))

**Output:**

Anushya

Auha

**String Order**

Write a program to accept an alphanumeric string and sort the characters of the string – first alphabets, followed by digits, followed by any other characters.

str = input()

alpha = []

digits = []

spl = []

for ch in str:

if ch.isaplha():

alpha.append(ch)

elif ch.isdigit():

digits.append(ch)

else:

spl.append(ch)

output = ‘’.join(sorted(alpha) + sorted(digits) + sorted(spl))

print(output)

**Output:**

Anu123@python

Ahnnoptuy123@

**String Reverse**

Write a Python program to reverse a String if the length of the String is a multiple of 4. And if it is not display -1 as the result.

string = input()

if len(string) % 4 == 0:

newStr = string[::-1]

print(newStr)

else:

print(“-1”)

**Output:**

Programs

smargorP

**String Mix-4**

Write a Python program to merge characters of two different string into a single string by taking characters alternatively.

If the strings are of unequal lengths, then all the remaining characters are to be put at the end of the new string.

str1 = input()

str2 = input()

i, j = 0, 0

output = “”

while i < len(str1) or j < len(str2):

if i < len(str1):

output = output + str1[i]

i = i + 1

if j < len(str2):

output = output +str2[j]

j = j + 1

print(output)

**Output:**

saianushya

srikanth

ssariiaknaunsthhya

**Even Odd**

Write a Python program to print characters present at even index and odd index positions.

str = input()

i = 0

while i < len(str):

print(str[i], end = ‘’)

i = i + 2

print()

i = 1

while i < len(str):

print(str[i], end = ‘’)

i = i + 2

**Output:**

Programming

Pormig

rgamn

**Forward and Backward**

Write a Python program to access each character of a string in both forward and backward directions by using the while loop.

s = input()

n = input()

i = 0

while i < n:

print(s[i], end = ‘’)

i += 1

print()

i = -1

while i >= -n:

print(s[i], end = ‘’)

i -= 1

**Output:**

P y t h o n

n o h t y P

**New Concatenated List**

Write a Python program to read two list values from the user and concatenate the list index wise. If the list is of unequal length, pair the values with a ‘-’.

from itertools impor zip\_longest as zl

list1 = eval(input())

list2 = eval(input())

list3 = []

for i, j in zl(list1, list2, fillvalue = ‘-’ ):

list3.append(i + j)

print(list3)

**Output:**

[‘A’ , ‘u’, ‘h’, ‘a’]

[‘n’, ‘s’, ‘y’]

[‘An’, ‘us’, ‘hy’, ‘a-’]

**Multiply within the list**

Write a Python program to multiply all the elements within the list and print the result.

def multiply\_list(items):

tot = 1

for x in items:

tot \*= x

return tot

l = eval(input())

print(multiply\_list(l))

**Output:**

[1, 2, -8]

-16

**Smallest number within the list**

Write a Python program to read a list dynamically and print the smallest number within the list.

def smallest\_num\_in\_list(list):

min = list[0]

for a in list:

if a < min:

min = a

return min

l = eval(input())

print(smallest\_num\_in\_list(l))

**Output:**

[17, 4, 9]

4

**Flatten a shallow list**

Write a Python program to flatten a shallow list.

l1 = eval(input())

l3 = []

for x in l1;

l3.extend(x)

print(l3)

**Output:**

[[3, 6, 3], [9, 13, 4], [17], [3, 56, 0]]

[3, 6, 3, 9, 13, 4, 17, 3, 56, 0]

**Position of n value**

Write a Python program to change the position of every n-th value with the (n + 1)th in a list.

from iteraools import zip\_longest, chain, tee

def replace2copy(lst):

lst1, lst2 = tee(iter(lst),2)

return list(chain.from\_iterable(zip\_longest(lst[1::2], lst[::2])))

n = eval(input())

print(replace2copy(n))

**Output:**

[0, 1, 2, 3, 4, 5]

[1, 0, 3, 2, 5, 4]

**Conversion of multiple integers**

Write a Python program to convert a list of multiple integers into a single integer.

L = eval(input())

x =int(“”.join(map(str, L)))

print(x)

**Output:**

[11, 33, 50]

113350

**Convert a pair of Values**

Write a Python program to convert a pair of values into a sorted unique array.

L = eval(input())

print(sorted(set().union(\*L)))

**Output:**

[(1, 2), (3, 4), (1, 2), (5, 6), (7, 8), (1, 2), (3, 4), (3, 4), (7, 8), (9, 10)]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

**Convert list to list of dictionaries**

Write a Python program to convert list to list of dictionaries.

color\_name = eval(input())

color\_code = eval(input())

r = []

for f, c in zip(color\_name, color\_code):

r.append(f: c{})

print(r)

**Output:**

[“Black”, “Red”, “Maroon”, “Yellow”]

[“#000000”, “#FF0000”, “#800000”, “#FFFF00”]

[{‘Black’: ‘#000000’}, {‘Red’: ‘#FF0000’}, {‘Maroon’: ‘#800000’}, {‘Yellow’: ‘#FFFF00’}]

**Iterate over two lists**

Write a Python program to iterate over two lists simultaneously.

l1 = eval(input())

l2 = eval(input())

for a, b in zip(l1, l2):

print(a, b)

**Output:**

[1, 2, 3]

[‘red’, ‘white’, ‘black’]

1 red

2 white

3 black

**Insert an element**

Write a Python program to insert an element at a specified position into a given list.

def insert\_spec\_position(x, n\_list, pos):

return n\_list[:pos – 1] + [x] + n\_list[pos – 1:]

# Reading the List dynamically

n\_list = eval(inout())

# Position to Insert the New Element

kth\_postion =int(input())

# New Element Value

x = eval(input())

result = insert\_spec\_position(x, n\_list, kth\_postion)

print(result)

**Output:**

[“A”, “B”, “C”, “D”, “E”]

1

‘a’

[‘a’, ‘A’, ‘B’, ‘C’, ‘D’, ‘E’]

**Split a given list**

Write a Python program to split a given list into two parts where the length of the first part of the list is given.

def split\_two\_parts(n\_list, L):

return n\_list[:L], n\_list[L:]

# Reading a List dynamically

n\_list = eval(input())

# No. of elements in List 1

first\_list\_length = int(input())

print(split\_two\_parts(n\_list, first\_list\_length))

**Output:**

[“C”, “C++”, “Java”, “Python”, “Ruby”, “JavaScript”, “HTML”, “CSS”]

5

([‘C’, ‘C++’, ‘Java’, ‘Python’, ‘Ruby’], [‘JavaScript’, ‘HTML’, ‘CSS’])

**Round the floating-point**

Write a Python program to round every number of a given list of floating-point values and print the total sum multiplied by the length of the list.

nums = eval(input())

# Finding the Length of the list

length = len(nums)

# Rounding off every element within the List

n1 = list(map(round, nums))

print(n1)

# Finding t0 Sum

# And Multiplying it with the Length of the List

s = sum((n1))

r = s \* length

# Printing the Result

print(s)

print(length)

print(r)

**Output:**

[22, 4, 4.0, -16.22, -9.10, 11.00, -12.22, 14.20, -5,.20, 17.50]

[22, 4, -16, -9, 11, -12, 14, -5, 18]

27

9

243

**Print the number**

Write a Python program to round the numbers of a given list, print the minimum and maximum numbers and multiply the numbers by 5.

Print the unique numbers in ascending order separated by space.

nums = eval(input())

# Rounding off each element inside the list

numbers = list(map(round, nums))

print(numbers)

# Finding the Minimum & Maximum Value

print(min(numbers))

print(max(numbers))

# Taking all the Unique Numbers from the rounded off List

numbers = list(set(numbers))

# Multiplying each element of the List with 5

# Sorting the List in ascending order

numbers = (sorted(map(lambda n: n \* 5, numbers)))

# Printing the elements of the Sorted List

for numb in numbers:

print(numb, end = ‘ ’)

**Output:**

[22.4, 4.0, 16.22, 9.10, 11.00, 12.22, 14.20, 5.20, 17.50]

[22, 4, 16, 9, 11, 12, 14, 5, 18]

4

22

20 25 45 55 60 70 80 90 110

**Zipping the values**

Write a Python program to Zip two given lists of lists.

# Dynamically Reading the List values

list1 = eval(input())

list2 =eval(input())

# Zipping the Lists

result = list(map(list.\_\_add\_\_, list1, list2))

# Printing the Zipped List

Print(result)

**Output:**

[[89, 23, 54, 25], [“Python”, “C++”]]

[78, 89], [“C”]]

[[89, 23, 54, 25, 78, 89], [“Python”, “C++”, “C”]]

**Maximum & Minimum**

Write a Python program to find the list with maximum and minimum length and value.

def max\_length\_list(input\_list):

max\_length = max(len(x) for x in input\_list)

max\_list = max(input\_list, key = len)

return(max\_length, max\_list)

def min\_length\_list(input\_list):

min\_length = min(len(x) for x in input\_list)

min\_list = min(input\_list, key = len)

return(min\_length, min\_list)

list1 = eval(input())

print((max\_length\_list(list1)))

print((min\_length\_list(list1)))

**Output:**

[[2, 4], [[6,8], [4, 5, 8]], [10, 12, 14]]

(3, [10, 12, 14])

(2, [2,4])

**Count the number**

Write a Python Program to count the number of sublists contain a given element.

**Note:** If the given element to search for is a String then while givin the input enclose the string within Single or Double Quotation.

def count\_element\_in\_list(input\_list, x):

ctr = 0

for i in range(len(input\_list):

if x in input\_list[i]:

ctr += 1

return ctr

list1 = eval(input())

ele = evel(input())

print(count\_element\_in\_list(list1, ele))

**Output:**

[[12], [1], [1, 3], [1, 34, 5, 7], [1, 34, 5, 17], [9, 11], [3, 5, 7]]

1

4

**Count number of sub-lists**

Write a Python program to count number of sub-lists within a given list.

def unique\_sublist(input\_list):

result = {}

for i in input\_list:

result.setdefault(tuple(l), list()).append(l)

for a, b in result.items():

result[a] = sum(b)

return result

list1 = eval(input())

print(unique\_sublists(list1))

**Output:**

[[1, 3], [5, 7], [1, 3], [13, 15, 17], [5, 7], [9, 11]]

{(1, 3): 2, (5, 7): 2, (13, 15, 17): 1, (9, 11): 1}

**List Square**

Write a Python program to read a list from the user and turn every item of a list into List square.

# Empty List

aList = []

rList = []

# Number of element of the list

n = int(input())

# Reading the value dynamically and inserting into the List

for i in range(n):

x = int(input())

aList.append(x)

# Calculating the Square

rList.append(x \* x)

# Printing the values

print(aList)

print(rList)

**Output:**

7

1

2

3

4

5

6

7

[1, 2, 3, 4, 5, 6, 7]

[1, 4, 9, 16, 25, 36, 49]

**New Concatenated List**

Write a Python program to read two lists from the user and concatenate them in the below given format.

list1 = eval(input())

list2 = eval(input())

list3 = []

for i in list1:

for j in list2:

list3.append(i + j)

print(list3)

**Output:**

[“Hello”, “take”]

[“Dear”, “Sir”]

[‘Hello Dear’, ‘Hello Sir’, ‘take Dear’, ‘take Sir’]

**Concatenating a given list**

Write a Python program to create a list by concatenating a given list which range goes from 1 to n.

my\_list = eval(input())

n = int(input())

r = []

for y in range(1, n + 1):

for x in my\_list:

r.append(‘{}{}’.format(x,y))

print(r)

**Output:**

[‘p’, ‘q’]

4

[‘p1’, ‘q1’, ‘p2’, ‘q2’, ‘p3’, ‘q3’, ‘p4’, ‘q4’]

**Iterate over two Lists**

Write a Python program to iterate over two lists simultaneously.

l1 = eval(input())

l2 = eval(input())

for a, b in zip(l1, l2):

print(a, b)

**Output:**

[1, 2, 3]

[‘red’, ‘white’, ‘black’]

1 red

2 white

3 black

**Insert an element at a specified position**

Write s Python to insert an element at a specified position into a given list.

def insert\_spec\_position(x, n\_list, pos):

return n\_list[:pos – 1] + [x] + n\_list[pos – 1:]

# Reading the list dynamically

n\_list = eval(input())

# Position to Insert the New Element

kth\_position = int(input())

# New Element Value

x = eval(input())

result = insert\_spec\_position(x, n\_list, kth\_position)

print(result)

**Output:**

[“A”, “B”, “C”, “D”, “E”]

1

‘a’

[‘a’, ‘A’, ‘B’, ‘C’, ‘D’, ‘E’]

**Split a given List**

Write a Python program to split a given list two parts where the length of the first part of the list is given.

def split\_two\_parts(n\_list, L):

return n\_list[:L], n\_list[L:]

# Reading a list dynamically

n\_list = eval(input())

# No. of elements in List 1

first\_list\_length = int(input())

print(split\_two\_parts(n\_list, first\_list, list\_length))

**Output:**

[“C”, “C++”, “Java”, “Python”, “Ruby”, “Javascript”, “HTML”, “CSS”]

5

([“C”, “C++”, “Java”, “Python”, “Ruby”], [“Javascript”, “HTML”, “CSS”])

**Round the Floating-Point**

Write a Python program to round every number of a given list of floating-point values and print the total sum multiplied by the length of the list.

nums = eval(input())

# Finding off every element within the list

n1 = list(map(round, nums))

print(n1)

# Finding the Sum

# And multiplying it with the Length of the List

s = sum((n1))

r = s \* length

# Printing the Result

print(s)

print(length)

print(r)

**Output:**

[22.4, 4.0, -16.22, -9.10, 11.00, -12.22, 14.20, -5.20, 17.50]

[22, 4, -16, -9, 11, -12, 14, -5, 18]

27

9

243

**Print the number**

Write a Python program to round the numbers of a given list, print the minimum and maximum numbers and multiply the numbers by 5.

Print the unique numbers in ascending order separated by space.

nums = eval(input())

# Rounding off each element inside the list

numbers = list(map(round, nums))

print(numbers)

# Finding the Minimum and Minimum Values

print(min(numbers)

print(max(numbers))

# Taking all the Unique Numbers from the rounded off List

numbers = list(set(numbers))

# Multiplying each element of the list with 5

# Sorting the list in ascending order

numbers = (sorted(map(lambda n:n \* 5, numbers)))

# Printing the elements of the Sorted List

for numb in numbers:

print(numb, end = ‘ ‘)

**Output:**

[22.4, 4.0, 16.22, 9.10, 11.00, 12.22, 14.20, 5.20, 17.50]

[22, 4, 16, 9, 11, 12, 14, 5, 18]

4

22

20 25 45 55 60 70 80 90 110

**Zipping the Values**

Write a Python program to Zip two given lists of lists.

# Dynamically Reading the List values

list1 = eval(input())

list2 = eval(input())

# Zipping the Lists

result = list(map(list.\_\_add\_\_, list1, list2))

# Printing the Zipping List

print(result)

**Output:**

[[89.23, 54.25], [“Python”, “C++”]]

[[78, 89], [“C”]]

[[89.23, 54.25, 78, 89], [“Python”, “C++”, “C”]]

**Minimum & Maximum**

Write a Python program to find the list with maximum and minimum length and value.

def max\_length\_list(input\_list):

max\_length = max(len(x) for x in input\_list)

max\_list = max(input\_list, key = len)

return(max\_length, max\_list)

def min\_length\_list(input\_list):

min\_length = min(len(x) for x in input\_list)

min\_list = min(input\_list, key = len)

return(min\_length, min\_list)

list1 = eval(input())

print((max\_length\_list(list1))

print((min\_length\_list(list1))

**Output:**

[[2, 4], [[6, 8], [4, 5, 8]], [10, 12, 14]]

(3, [10, 12, 14])

(2, [2, 4])

**Count the number of sub-lists with given elements**

Write a Python program to count the number of sub-lists contain a given element.

def count\_element\_in\_list(input\_list, x):

ctr = 0

for i in range(len(input\_list)):

if x in input\_list[i]:

str += 1

return ctr

list1 = eval(input())

ele = eval(input())

print(count\_element\_in\_list(list1, ele))

**Output:**

[[12], [1], [1, 3], [1, 34, 5, 17], [9, 11], [3, 5, 7]]

1

4

**Count number of sub-list with given list.**

Write a Python program to count number of sublists within a given list.

def unique\_sublists(input\_l;ist):

result = {}

for l in input\_list:

result.setdefault(tuple(l), list()).append(l)

for a, b in result.items():

result[a] = sum(b)

return result

list1 = eval(input())

print(unique\_sublists(list1))

**Output:**

[[1, 3], [5, 7], [1, 3], [13, 15, 17], [5, 7], [9, 11]]

{(1, 3) : 2, (5, 7) : 2, (13, 15, 17) : 1, (9, 11) : 1}

**List Square**

Write a Python program to read a list from the user and turn every item of a list into List Square.

# Empty List

aList = []

rList = []

# Number of element of the list

n = int(input())

# Reading the value dynamically and inserting into the List

for i in range(n):

x = int(input())

aList.append(x)

# Calculating the Square

rList.append(x \* x)

# Printing the values

print(aList)

print(rList)

**Output:**

7

1

2

3

4

5

6

7

[1, 2, 3, 4, 5, 6, 7]

[1, 4, 9, 16, 25, 36, 49]

**New Concatenated List**

Write a Python program to read two lists from the user and concatenate them in the below given format.

list1 = eval(input())  
list2 = eval(input())

list3 = []

for i in list1:

for j in list2:

list3.append(i + j)

print(list3)

**Output:**

[“Hello”, “take”]

[“Dear”, “Sir”]

[‘Hello Dear’, ‘Hello Sir’, ‘take Dear’, ‘take Sir]

**Concatenating a given List**

Write a Python program to create a list by concatenating a given list which range goes from 1 to n.

my\_list = eval(input())

n = int(input())

r = []

for y in range(1, n + 1):

for x in my\_list:

r.append(‘{}{}’.format(x, y))

print(r)

**Output:**

[‘p’, ‘q’]

4

[‘p1’, ‘q1’, ‘p2’, ‘q2’, ‘p3’, ‘q3’, ‘p4’, ‘q4]

**Iterate over two List**

Write a Python program to read 2 lists from the user and iterate over both the lists simultaneously such that list1 should display item in the original order and list2 in reverse order.

list1 = eval(input())

list2 = eval(input())

for x, y in zip(list1, list2[::-1]):

print(x, y)

**Output:**

[10, 20, 30, 40]

[100, 200, 300, 400]

10 400

20 300

30 200

40 100

**Return True if First and Last Same**

Given a list integers, return True if first and last number of a list is same.

List = eval(input())

f = List[0]

l = List[-1]

if(f == l):

print(“1”)

else:

print(“0”)

**Output:**[10, 20, 30, 40, 10]

1

**Remove all the Empty Strings**

Write a Python program to read a list dynamically and remove all the empty strings from the list and print it.

list1 = eval(input())

result = list(filter(None, list1))

print(result)

**Output:**

[“Mike”, “”, “Emma”, “Kelly”, “”, “Brad”]

[‘Mike’, ‘Emma’, ‘Brad’]

**Replace the value**

Write a Python program to read a list dynamically. Find value 20 in the list, and if it is present, replace it with 200. Only update the first occurrence of a value. If the value is not present, return the list as it is.

list1 = eval(input())

if 20 in list1:

index = list1.index(20)

list1[index] = 200

print(list1)

else:

print(list1)

**Output:**

[5, 10, 15, 25, 50, 20]

[5, 10, 15, 25, 50, 200]

**Put even and odd**

Write a Python program to read a list dynamically and put the even and odd elements in it into separate lists.

a = []

n = int(input())

for I in range(1, n + 1):

b = int(input())

a.append(b)

even = []

odd = []

for j in a:

if(j % 2 == 0):

even.append(j)

else:

odd.append(j)

print(even)

print(odd)

**Output:**

Enter number of elements: 5

Enter element: 67

Enter element: 43

Enter element: 44

Enter element: 22

Enter element: 455

The even list [44, 22]

The odd list [67, 43, 455]

**Find the union of two lists**

Write a Python a program to find the union of two lists. And print the sorted list as the result.

l1 = []

num1 = int(input())

for n in range(num1):

numbers1 = int(input())

l1.append(numbers1)

l2 = []

num2 = int(input())

for n in range(num2):

numbers2 = int(input())

l2.append(numbers2)

l3 = []

l3.extend(l1)

for i in l2:

if i in l1:

pass

else:

l3.append(i)

l3.sort()

print(l3)

**Output:**

4

20

40

30

60

4

10

20

50

40

[10, 20, 30, 40, 50, 60]

**Intersection of lists**

Write a Python program to read 2 lists dynamically from the user and find the intersection of the two lists in a sorted order.

l1 = []

l2 = []

num1 = int(input())

num2 = int(input())

for n in range(num1):

numbers1 = int(input())

l1.append(numbers1)

for n in range(num2):

numbers2 = int(input())

l1.append(numbers2)

l3 = []

for i in l2:

if i in l1:

l3.append(i)

l3.sort()

print(l3)

**Output:**

3

4

34

23

65

33

65

23

86

[23, 65]

**List after remove**

Write s Python program to dynamically read a list and remove all the occurrences of 20 from the list and print the resultant list.If 20 is not present, then print the list as it is.

def removeValue(sampleList, val):

if 20 in list1:

return [Value for value in sampleList if value != val]

else:

return sampleList

list1 = eval(input())

resList = removeValue(list1, 20)

print(resList)

**Output:**

[5, 20, 15, 20, 25, 50, 20]

[5, 15, 25, 50]

**Even and Odd number**

Given a two list of numbers, create a third list such that should contain only odd numbers from the first list and even numbers from the second list.

One = eval(input())

Two = eval(input())

Three = []

for num in One:

if(num % 2 != 0):

Three.append(num)

for num in Two:

if(num % 2 != 0):

Three.append(num)

print(Three)

**Output:**

[25, 78, 90, 58, 10]

[85, 25, 30, 40, 87, 90]

[25, 30, 40, 90]

**Largest number within list**

Write a Python program to dynamically read a list and find the largest number within the list.

a = []

n = int(input())

for i in range(n):

b = int(input())

a.append(b)

a.sort()

print(a[n -1])

**Output:**

3

23

567

3

567

**Find second large**

Write a Python program to read a list dynamically and find the 2nd largest number within the given list.

a = []

n = int(input())

for I in range(l, n + 1):

b = int(input())

a.append(b)

a.sort()

print(a[n – 2])

**Output:**

4

23

56

39

11

39

**Interchange first and last**

Write a Python program to interchange the first and last elemant of the List

List\_1 = eval(input())

n = len(List\_1)

temp = List\_1[0]

List\_1[0] = List\_1[n-1]

List\_1[n – 1] = temp

Print(List\_1)

**Output:**

[5, 10, 15, 20, 25, 30, 35, 40]

[40, 10, 15, 20, 25, 30, 35, 5]

**Length of the string**

Write a Python program to read a list of strings and print the count of the strings whose length is greater than 2 and also the first and last character of the string is same.

def match\_words(words):

ctr = 0

for w in words:

if len(w) > 2 and w[0] == w[-1]:

ctr +=1

return ctr

l = eval(input())

print(match\_words(l))

**Output:**

[‘abc’, ‘knok’, ‘kick’, ‘java’, ‘aa’, ‘bb’]

3

**Duplicate values**

Write a Python program to remove all the duplicate values from the list and print the resultant list in sorted order.

a = eval(input())

s = set(a)

a = list(s)

a.sort()

print(a)

**Output:**

[10, 20, 52, 10, 20, 55, 52, 78, 96]

[10, 20, 52, 55, 78, 96]

**Boolean Value**

Write a Python program to take two lists from the user and return True if at least one matching element is present between the two lists.

def common\_data(list1, list2):

result = False

for x in list1:

for y in list2:

if x == y:

result = True

return result

L1 = eval(input())

L2 = eval(input())

print(common\_data(L!, L2))

**Output:**

[1, 2, 3, 4, 5]

[5, 6, 7, 8, 9]

True

**Remove all the element**

Write a Python program to remove all the even numbers from the list and print the resultant list.

num = eval(input())

num = [x for c in num if x % 2 != 0]

print(num)

**Output:**

[7, 8, 120, 25, 44, 20, 27]

[7, 25, 27]

**Square of the number**

Write a Python program to create a list with its 1st element as a number and the 2nd element as the square of that number.

l = int(input())

u = int(input())

t = []

for x in range(l, u + 1):

a = [x, x \*\* 2]

t.append(a)

print(t)

**Output:**

1

4

[[1, 1], [2, 4], [3, 9], [4, 16]]

**Length of the longest word**

Write a Python program to rad a list of words and return the length of the longest word present within the list elements.

a = []

n = int(input())

for x in range(0, n):

element = input()

a.append(element)

max1 = len(a[0])

temp = a[0]

for i in a:

if (len(i) > max1):

max1 = len(i)

temp = i

print(temp)

**Output:**

4

pie

apple

orange

pineapple

pineapple