RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

CS23221
PYTHON PROGRAMMING LAB

Laboratory Record Note Book

Name : Sanjay kumar.k

Year / Branch / Section : 1st/CSE/"E"

Register No. : 230701289

Semester : 2nd

Academic Year : 2023-2024

RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name : . . . . .SANJAY KUMAR.K . . . .
. .

Academic Year : 2023-.2024 . . . . Semester : . .2ND . . . .
Branch : . . . . CSE. . . .
. .

Register No. 230701289

Certified that this is the bonafide record of work done by the above student

in the ....PYTHON PROGRAMMING LABORATORY ....

. . . . . . Laboratory during the year 2023 - 2024

Signature of Faculty in-charge

Submitted for the Practical Examination held on . . . .

Internal Examiner **External Examiner**

INDEX

Reg. No. : 230701289 Name : SANJAY KUMAR.K

Year : 2023-2024 Branch : CSE Sec : "E"

01 - Introduction to Python-Variables-Datatypes Input/Output-Formatting

Sample Output:

```
10,<class 'int'>
10.9,<class 'float'>
```

For example:

Input	Result
10 10.9	10,<class 'int'> 10.9,<class 'float'>

Ex. No. : 1.1 **Date:**

Register No.:230701289 **Name:** SANJAY KUMAR.K

Converting Input Strings

Write a program to convert strings to an integer and float and display its type.

Sample Input:

10

10.9

Program:

```
val = int(input())

print(f"{val},{type(val)}")

val = float(input())

print(f"{val:.1f},{type(val)}")
```

Sample Input:

10000

Sample Output:

16000

For example:

Input	Result
10000	16000

Ex. No. : 1.2 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Gross Salary

Ramesh’s basic salary is input through the keyboard. His dearness allowance is 40% of his basic salary, and his house rent allowance is 20% of his basic salary. Write a program to calculate his gross salary.

Program:

```
basicPay = int(input())

print(int(basicPay + basicPay * .6))
```

Sample Input:

8.00

Sample Output:

2.828

For example:

Input	Result
14.00	3.742

Ex. No. : 1.3 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Square Root

Write a simple python program to find the square root of a given floating point number. The output should be displayed with 3 decimal places.

Program:

```
from math import sqrt

sqroot = sqrt(float(input()))

print(f"{sqroot:.3f}")
```

.

Input Format:

The first line contains the Rs X

The second line contains Rs Y

The third line contains Rs Z

Sample Input:

10000

250

15000

Sample Output:

46.34 is the gain percent.

For example:

Input	Result
45500 500 60000	30.43 is the gain percent.

Ex. No. : **1.4** **Date:**

Register No.: **:230701289** **Name:** sanjay kumar.k

Gain percent

Alfred buys an old scooter for Rs. X and spends Rs. Y on its repairs. If he sells the scooter for Rs. Z (Z>X+Y). Write a program to help Alfred to find his gain percent. Get all the above-mentioned values through the keyboard and find the gain percent.

Program:

```
x = int(input())

y = int(input())

z = int(input())

print(f"{((z - (x + y)) / (x + y) ) * 100:.2f} is the gain percent.")
```

Sample Input

10

20

Sample Output

Your total refund will be $6.00.

For example:

Input	Result
20 20	Your total refund will be $7.00.

Ex. No. : 1.5 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Deposits

In many jurisdictions, a small deposit is added to drink containers to encourage people to recycle them. In one particular jurisdiction, drink containers holding one liter or less have a $0.10 deposit and drink containers holding more than one liter have a $0.25 deposit. Write a program that reads the number of containers of each size(less and more) from the user. Your program should continue by computing and displaying the refund that will be received for returning those containers. Format the output so that it includes a dollar sign and always displays exactly two decimal places.

Program:

```
print(f"Your total refund will be ${(0.1 * int(input())) + (.25 * int(input())):.2f}.")
```

Sample Input:

```
450
```

Sample Output:

```
weekdays 10.38
weekend 0.38
```

For example:

Input	Result
450	weekdays 10.38 weekend 0.38

Ex. No. : 1.6 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Carpenter

Justin is a carpenter who works on an hourly basis. He works in a company where he is paid Rs 50 for an hour on weekdays and Rs 80 for an hour on weekends. He works 10 hrs more on weekdays than weekends. If the salary paid for him is given, write a program to find the number of hours he has worked on weekdays and weekends.

Hint:

If the final result(hrs) are in -ve convert that to +ve using abs() function

The abs() function returns the absolute value of the given number.

```
number = -20
absolute_number = abs(number)
print(absolute_number)
# Output: 20
```

Program:

```
sal = int(input())

days = abs( (sal - 500) / 130)

print(f"weekdays {days + 10:.2f}")
print(f"weekend {days:.2f}")
```

02- Operators in Python

Sample Input

10

20

Sample Output

The total weight of all these widgets and gizmos is 2990 grams.

For example:

Input	Result
10 20	The total weight of all these widgets and gizmos is 2990 grams.

Ex. No. : 2.1 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Widgets and Gizmos

An online retailer sells two products: widgets and gizmos. Each widget weighs 75 grams. Each gizmo weighs 112 grams. Write a program that reads the number of widgets and the number of gizmos from the user. Then your program should compute and display the total weight of the parts.

Program:

```
print(f"The total weight of all these widgets and gizmos is {int(input()) * 75 + int(input()) *
112} grams.")
```

Sample Input

10

Sample Output

True

Explanation:

Since 10 is an even number and a number between 0 and 100, True is printed

.

Ex. No. : 2.2 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Doll Sings

In London, every year during Dasara there will be a very grand doll show. People try to invent new dolls of different varieties. The best-sold doll's creator will be awarded with a cash prize. So people broke their heads to create dolls innovatively. Knowing this competition, Mr.Lokpaul tried to create a doll that sings only when an even number is pressed and the number should not be zero and greater than 100.

IF Lokpaul wins print true, otherwise false.

Program:

```
num = int(input())

result = num > 0 and (num & 1 == 0) and num < 100

print(result)
```

Input Given:

N-No of friends

P1,P2,P3 AND P4-No of chocolates

OUTPUT:

"True" if he can buy that packet and "False" if he can't buy that packet.

SAMPLE INPUT AND OUTPUT:

```
5
25
12
10
9
```

OUTPUT

```
True False True False
```

Ex. No. : 2.3 Date:

Register No.: :230701289 Name: sanjay kumar.k

Birthday Party

Mr. X's birthday is in next month. This time he is planning to invite N of his friends. He wants to distribute some chocolates to all of his friends after the party. He went to a shop to buy a packet of chocolates. At the chocolate shop, 4 packets are there with different numbers of chocolates. He wants to buy such a packet which contains a number of chocolates, which can be distributed equally among all of his friends. Help Mr. X to buy such a packet.

Program:

```
num = int(input())
count = int(input())
print(count % num == 0, end = " ")
count = int(input())
print(count % num == 0, end = " ")
count = int(input())
print(count % num == 0, end = " ")
count = int(input())
print(count % num == 0, end = " ")
```

Sample Input

3

Sample Output:

2

Explanation:

The binary representation of 3 is 011, hence there are 2 ones in it. so the output is 2.

Ex. No. : 2.4 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Hamming Weight

Write a python program that takes a integer between 0 and 15 as input and displays the number of '1' s in its binary form.(Hint:use python bitwise operator.

Program:

```
num = int(input())

mask = 1

count = 0

count += num & mask

num >>= 1

count += num & mask

num >>= 1

count += num & mask

num >>= 1

count += num & mask

num >>= 1

print(count)
```

Sample Input:

10000

Sample Output:

Balance as of end of Year 1: $10400.00.

Balance as of end of Year 2: $10816.00.

Balance as of end of Year 3: $11248.64

Ex. No. : 2.5 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Compound Interest

Pretend that you have just opened a new savings account that earns 4 percent interest per year. The interest that you earn is paid at the end of the year, and is added to the balance of the savings account. Write a program that begins by reading the amount of money deposited into the account from the user. Then your program should compute and display the amount in the savings account after 1, 2, and 3 years. Display each amount so that it is rounded to 2 decimal places.

.

Program:

```
basic = int(input())

basic += basic * .04

print(f"Balance as of end of Year 1: ${basic:.2f}.")

basic += basic * .04

print(f"Balance as of end of Year 2: ${basic:.2f}.")

basic += basic * .04

print(f"Balance as of end of Year 3: ${basic:.2f}.")
```

.

Input Format:

Input consists of two integers that correspond to the age and weight of a person respectively.

Output Format:

Display True(IF ELIGIBLE)

Display False (if not eligible)

Sample Input

19

45

Sample Output

True

Ex. No. : 2.6 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Eligible to donate blood

A team from the Rotract club had planned to conduct a rally to create awareness among the Coimbatore people to donate blood. They conducted the rally successfully. Many of the Coimbatore people realized it and came forward to donate their blood to nearby blood banks. The eligibility criteria for donating blood are people should be above or equal to 18 and his/ her weight should be above 40. There was a huge crowd and staff in the blood bank found it difficult to manage the crowd. So they decided to keep a system and ask the people to enter their age and weight in the system. If a person is eligible he/she will be allowed inside.

Write a program and feed it to the system to find whether a person is eligible or not.

Program:

```
age = int(input())

weight = int(input())

canDonate = age >= 18 and weight > 40

print(canDonate)
```

Input Format:

An integer x, 0<=x<=1. .

Output Format:

output a single character "C" or "D"depending on the value of x.

Input 1:
0
Output 1:

C

Input 2:
1

Output 1:
D

Ex. No. : 2.7 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

C or D

Mr.Ram has been given a problem kindly help him to solve it. The input of the program is either 0 or 1. IF 0 is the input he should display "C" if 1 is the input it should display "D".There is a constraint that Mr. Ram should use either logical operators or arithmetic operators to solve the problem, not anything else.

Hint:

Use ASCII values of C and D.

Program:

```
print(chr(ord("C") + int(input())))
```

.

Input format:

Line 1 has the total number of weapons

Line 2 has the total number of Soldiers.

Output Format:

If the battle can be won print True otherwise print False.

Sample Input:

32

43

Sample Output:'

False

Ex. No. : 2.8 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Troy Battle

In the 1800s, the battle of Troy was led by Hercules. He was a superstitious person. He believed that his crew can win the battle only if the total count of the weapons in hand is in multiple of 3 and the soldiers are in an even number of count. Given the total number of weapons and the soldier's count, Find whether the battle can be won or not according to Hercules's belief. If the battle can be won print True otherwise print False.

Program:

```
weapons = int(input())
soldiers = int(input())
res = weapons % 3 == 0 and soldiers % 2 == 0
print(res)
```

.

Sample Input

100

Sample Output

The tax is 5.00 and the tip is 18.00, making the total 123.00

Ex. No. : 2.9 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Tax and Tip

The program that you create for this exercise will begin by reading the cost of a meal ordered at a restaurant from the user. Then your program will compute the tax and tip for the meal. Use your local tax rate (5 percent) when computing the amount of tax owing. Compute the tip as 18 percent of the meal amount (without the tax). The output from your program should include the tax amount, the tip amount, and the grand total for the meal including both the tax and the tip. Format the output so that all of the values are displayed using two decimal places.

Program:

```
cost = int(input())
print(f"The tax is {.05 * cost:.2f} and the tip is {.18 * cost:.2f}, making the total {(cost + cost * .23):.2f}")
```

For example:

Input	Result
123	3

Ex. No. : 2.10 **Date:**

Register No.: :230701289 **Name:** sanjay kumar.k

Return last digit of the given number

Write a program that returns the last digit of the given number. Last digit is being referred to the least significant digit i.e. the digit in the ones (units) place in the given number.

The last digit should be returned as a positive number.

For example,

if the given number is 197, the last digit is 7

if the given number is -197, the last digit is 7

Program:

```
print(abs(int(input())) % 10)
```

03 - Selection Structures in Python

Sample Test Cases

Test Case 1

Input

70

60

80

Output

The candidate is eligible

Test Case 2

Input

50

80

80

Output

The candidate is eligible

Test Case 3

Input

50

60

40

Output

The candidate is not eligible

For example:

Input	Result
50 80 80	The candidate is eligible

Ex. No. : 3.1 Date:

Register No.: 230701289 Name: sanjay kumar.k

Admission Eligibility

Write a program to find the eligibility of admission for a professional course based on the following criteria:

Marks in Maths >= 65

Marks in Physics >= 55

Marks in Chemistry >= 50

Or

Total in all three subjects >= 180

Program:

```
m, p, c = int(input()), int(input()), int(input())

if m + p + c >= 180 or (m >= 65 and p >= 55 and c >= 50):

    print("The candidate is eligible")

else:

    print("The candidate is not eligible")
```

Sample Input 1

60

60

60

Sample Output 1

That's a equilateral triangle

For example:

Input	Result
40 40 80	That's a isosceles triangle

Ex. No. : 3.2 **Date:**

Register No.: 230701289 sanjay kumar.k

Classifying Triangles

A triangle can be classified based on the lengths of its sides as equilateral, isosceles or scalene. All three sides of an equilateral triangle have the same length. An isosceles triangle has two sides that are the same length, and a third side that is a different length. If all of the sides have different lengths then the triangle is scalene.

Write a program that reads the lengths of the three sides of a triangle from the user. Then display a message that states the triangle's type.

Program:

```
sides = set([int(input()), int(input()), int(input())])
if len(sides) == 1:
   print("That's a equilateral triangle")
elif len(sides) == 2:
   print("That's a isosceles triangle")
else:
   print("That's a scalene triangle")
```

Sample Test Cases

Test Case 1

Input

50

Output

100.00

Test Case 2

Input

300

Output

517.50

For example:

Input	Result
500	1035.00

Ex. No. : 3.3 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Electricity Bill

Write a program to calculate and print the Electricity bill where the unit consumed by the user is given from test case. It prints the total amount the customer has to pay. The charge are as follows:

Unit	Charge / Unit
Upto 199	@1.20
200 and above but less than 400	@1.50
400 and above but less than 600	@1.80
600 and above	@2.00

If bill exceeds Rs.400 then a surcharge of 15% will be charged and the minimum bill should be of Rs.100/-

Program:

```
units = eval(input())
bill = 0
if units in range(200):
   bill = units * 1.20
elif units in range(200,400):
   bill = units * 1.50
elif units in range(400,600):
   bill = units * 1.80
else:
   bill = units * 2.00

if bill < 100:
   bill = 100
if bill > 400:
   bill += bill * 0.15

print(f"{bill:.2f}")
```

Input Format:

Input consists of 2 integers.

The first integer corresponds to the number of problems given and the second integer corresponds to the number of problems solved.

Output Format:

Output consists of the string “IN” or “OUT”.

Sample Input and Output:

Input

8

3

Output

OUT

For example:

Input	Result
8 3	OUT

Ex. No. **:** **3.4** **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

IN/OUT

Ms. Sita, the faculty handling programming lab for you is very strict. Your seniors have told you that she will not allow you to enter the week's lab if you have not completed atleast half the number of problems given last week. Many of you didn't understand this statement and so they requested the good programmers from your batch to write a program to find whether a student will be allowed into a week's lab given the number of problems given last week and the number of problems solved by the student in that week.

Program:

```
problems, solved = int(input()) / 2, int(input())
if solved >= problems:
   print("IN")
else:
   print("OUT")
```

Sample Input 1

i

Sample Output 1

It's a vowel.

Sample Input 2

y

Sample Output 2

Sometimes it's a vowel... Sometimes it's a consonant.

Sample Input3

c

Sample Output 3

It's a consonant.

For example:

Input	Result
y	Sometimes it's a vowel... Sometimes it's a consonant.
u	It's a vowel.
p	It's a consonant.

Ex. No. : 3.5 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Vowel or Consonant

In this exercise you will create a program that reads a letter of the alphabet from the user. If the user enters a, e, i, o or u then your program should display a message indicating that the entered letter is a vowel. If the user enters 'y' then your program should display a message indicating that sometimes y is a vowel, and sometimes y is a consonant. Otherwise your program should display a message indicating that the letter is a consonant.

Program:

```
a = input()
if a in 'aeiou':
   print("It's a vowel.")
elif a == 'y':
   print("Sometimes it's a vowel... Sometimes it's a consonant.")
else:
   print("It's a consonant.")
```

Sample Input 1

1900

Sample Output 1

1900 is not a leap year.

Sample Input 2

2000

Sample Output 2

2000 is a leap year.

Ex. No. : 3.6 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Leap Year

Most years have 365 days. However, the time required for the Earth to orbit the Sun is actually slightly more than that. As a result, an extra day, February 29, is included in some years to correct for this difference. Such years are referred to as leap years. The rules for determining whether or not a year is a leap year follow:

- Any year that is divisible by 400 is a leap year.
- Of the remaining years, any year that is divisible by 100 is not a leap year.
- Of the remaining years, any year that is divisible by 4 is a leap year.
- All other years are not leap years.

Write a program that reads a year from the user and displays a message indicating whether or not it is a leap year.

Program:

```
year = int(input())

if year % 400 == 0:

   print(f"{year} is a leap year.")

else:

   if year % 100 != 0 and year % 4 == 0:

      print(f"{year} is a leap year.")

   else:

      print(f"{year} is not a leap year.")
```

Sample Input 1

February

Sample Output 1

February has 28 or 29 days in it.

Sample Input 2

March

Sample Output 2

March has 31 days in it.

Sample Input 3

April

Sample Output 3

April has 30 days in it.

For example:

Input	Result
February	February has 28 or 29 days in it.
March	March has 31 days in it.

Ex. No. : 3.7 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Month name to days

The length of a month varies from 28 to 31 days. In this exercise you will create a program that reads the name of a month from the user as a string. Then your program should display the number of days in that month. Display “28 or 29 days” for February so that leap years are addressed.

Program:

```
thirtyOnes = ["January", "March", "May", "July", "August", "October", "December"]

month = input().strip()

if month == "February":

    print(f"{month} has 28 or 29 days in it.")

elif month in thirtyOnes:

    print(f"{month} has 31 days in it.")

else:

    print(f"{month} has 30 days in it.")
```

Sample Input

3

5

4

Sample Output

Yes

For example:

Input	Result
3 4 5	Yes

Ex. No. : **3.8** **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Pythagorean triple

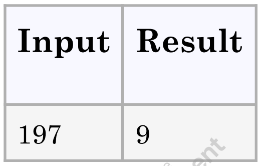
Three numbers form a Pythagorean triple if the sum of squares of two numbers is equal to the square of the third.

For example, 3, 5 and 4 form a Pythagorean triple, since 3*3 + 4*4 = 25 = 5*5
You are given three integers, a, b, and c. They need not be given in increasing order. If they form a Pythagorean triple, then print "Yes", otherwise, print "No".

Program:

```
vars = sorted([int(input()), int(input()), int(input())])
if vars[2] ** 2 == vars[0] ** 2 + vars[1] ** 2 :
   print("yes")
else:
   print("no")
```

For example:



Input	Result
197	9

Ex. No.	**:**	**3.9**	**Date:**	
Register No.:		230701289	**Name:**	sanjay kumar.k

Second last digit

Write a program that returns the second last digit of the given number. Second last digit is being referred 10the digit in the tens place in the given number.

For example, if the given number is 197, the second last digit is 9.

Note1 - The second last digit should be returned as a positive number. i.e. if the given number is -197, the second last digit is 9.

Note2 - If the given number is a single digit number, then the second last digit does not exist. In such cases, the program should return -1. i.e. if the given number is 5, the second last digit should be returned as -1.

Program:

```
num = abs(int(input()))

if num < 10:

    print(-1)

else:

    print((num % 100) // 10)
```

Sample Input 1

2010

Sample Output 1

2010 is the year of the Tiger.

Sample Input 2

2020

Sample Output 2

2020 is the year of the Rat.

Ex. No. : 3.10 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Chinese Zodiac

The Chinese zodiac assigns animals to years in a 12 year cycle. One 12 year cycle is shown in the table below. The pattern repeats from there, with 2012 being another year of the dragon, and 1999 being another year of the hare.

Year	Animal
2000	Dragon
2001	Snake
2002	Horse
2003	Sheep
2004	Monkey
2005	Rooster
2006	Dog
2007	Pig
2008	Rat
2009	Ox
2010	Tiger
2011	Hare

Write a program that reads a year from the user and displays the animal associated with that year. Your program should work correctly for any year greater than or equal to zero, not just the ones listed in the table.

Program:

```
year = int(input())

animal = ""

if year % 12 == 0: animal = "Monkey"
elif year % 12 == 1 : animal = "Rooster"
elif year % 12 == 2 : animal = "Dog"
elif year % 12 == 3 : animal = "Pig"
elif year % 12 == 4 : animal = "Rat"
elif year % 12 == 5 : animal = "Ox"
elif year % 12 == 6 : animal = "Tiger"
elif year % 12 == 7 : animal = "Hare"
elif year % 12 == 8 : animal = "Dragon"
elif year % 12 == 9 : animal = "Snake"
elif year % 12 == 10: animal = "Horse"
elif year % 12 == 11: animal = "Sheep"

print(f"{year} is the year of the {animal}.")
```

04 - Iteration Control Structures

For example:

Input	Result	
20	1 2 4 5 10 20	

Ex. No. : 4.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Factors of a number

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number).

Program:

```
number = int(input())
i = 1
while i <= number:
    # If the current number divides the target number evenly, it is a factor
    if number % i == 0:
        print(i)
    i += 1
```

For example:

Input	Result
292	1
1015	2
108	3
22	0

Ex. No. : **4.2** **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Non Repeated Digit Count

Write a program to find the count of non-repeated digits in a given number N. The number will be passed to the program as an input of type int.
Assumption: The input number will be a positive integer number >= 1 and <= 25000.
Some examples are as below.
If the given number is 292, the program should return 1 because there is only 1 non-repeated digit '9' in this number
If the given number is 1015, the program should return 2 because there are 2 non-repeated digits in this number, '0', and '5'.
If the given number is 108, the program should return 3 because there are 3 non-repeated digits in this number, '1', '0', and '8'.
If the given number is 22, the function should return 0 because there are NO non-repeated digits in this number.

Program:

```
N = int(input())
non_repeated_count = 0

for digit in range(10):
    found = False
    repeat = False
    temp = N
    while temp > 0:
        current_digit = temp % 10
        if current_digit == digit:
            if found:
                repeat = True
                break
            found = True
        temp //= 10
    if found and not repeat:
        non_repeated_count += 1
print(non_repeated_count)
```

.

Example1: if the given number N is 7, the method must return 2

Example2: if the given number N is 10, the method must return 1

For example:

Input	Result
7	2
10	1

Ex. No. :	**4.3**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Prime Checking

Write a program that finds whether the given number N is Prime or not. If the number is prime, the program should return 2 else it must return 1.

Assumption: 2 <= N <=5000, where N is the given number.

Program:

```
N = int(input())
is_prime = 2
if N == 2:
   is_prime = 2
elif N % 2 == 0:
   is_prime = 1
else:
   for i in range(3, int(N**0.5) + 1, 2):
      if N % i == 0:
         is_prime = 1  # Not prime
         break
print(is_prime)
```

Input Format:

Integer input from stdin.

Output Format:

Perfect square greater than N.

Example Input:

10

Output:

16

Ex. No. : 4.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Next Perfect Square

Given a number N, find the next perfect square greater than N.

Program:

```
import math

N = int(input())

next_int = math.ceil(math.sqrt(N))

next_perfect_square = next_int ** 2

print(next_perfect_square)
```

NOTE: Fibonacci series looks like –

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, . . . and so on.

i.e. Fibonacci series starts with 0 and 1, and continues generating the next number as the sum of the previous two numbers.

- first Fibonacci number is 0,
- second Fibonacci number is 1,
- third Fibonacci number is 1,
- fourth Fibonacci number is 2,
- fifth Fibonacci number is 3,
- sixth Fibonacci number is 5,
- seventh Fibonacci number is 8, and so on.

For example:

Input:

7

Output

8

Ex. No. : 4.5 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Nth Fibonacci

Write a program to return the nth number in the fibonacci series. The value of N will be passed to the program as input.

Program:

```
N = int(input())

a, b = 0, 1

if N == 1:

    print(a)

elif N == 2:

    print(b)

else:

    for _ in range(2, N):

        a, b = b, a + b

    print(b)  # b is the nth Fibonacci number
```

Input Format:

Single Integer Input from stdin.

Output Format:

Yes or No.

Example Input:

175

Output:

Yes

Explanation

1^1 + 7^2 +5^3 = 175

Example Input:

123

Output:

No

For example:

Input	Result
175	Yes
123	No

Ex. No. : 4.6 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Disarium Number

A Number is said to be Disarium number when the sum of its digit raised to the power of their respective positions becomes equal to the number itself. Write a program to print number is Disarium or not.

Program:

```
N = int(input())

temp = N

num_digits = 0

while temp > 0:

   num_digits += 1

   temp //= 10

sum_of_powers = 0

temp = N

while temp > 0:

   digit = temp % 10

   sum_of_powers += digit ** num_digits

   num_digits -= 1

   temp //= 10

if sum_of_powers == N:

   print("The number is a Disarium number.")

else:

   print("The number is not a Disarium number.")
```

Sample Test Cases

Test Case 1

Input

4

Output

1234

Explanation:

as input is 4, have to take 4 terms.

1 + 11 + 111 + 1111

Test Case 2

Input

6

Output

123456

For example:

Input	Result
3	123

Ex. No. :	**4.7**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Sum of Series

Write a program to find the sum of the series 1 +11 + 111 + 1111 + . . . + n terms (n will be given as input from the user and sum will be the output)

Program:

```
n = int(input())
term = 0
sum_of_series = 0
for i in range(1, n+1):
   term = term * 10 + 1
   sum_of_series += term
print(sum_of_series)
```

For example:

Input	Result
292	2
1015	3

Ex. No. :	4.8	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Unique Digit Count

Write a program to find the count of unique digits in a given number N. The number will be passed to the program as an input of type int.

Assumption: The input number will be a positive integer number >= 1 and <= 25000.

For e.g.

If the given number is 292, the program should return 2 because there are only 2 unique digits '2' and '9' in this number

If the given number is 1015, the program should return 3 because there are 3 unique digits in this number, '1', '0', and '5'.

Program:

```
N = int(input())
seen_once = 0
seen_multiple = 0
while N > 0:
    digit = N % 10
    digit_mask = 1 << digit
    if seen_once & digit_mask:
        seen_multiple |= digit_mask
    seen_once |= digit_mask
    N //= 10
non_repeated_digits = seen_once & ~seen_multiple
non_repeated_count = 0
for i in range(10):
    if non_repeated_digits & (1 << i):
        non_repeated_count += 1
print(non_repeated_count)
```

Input Format:

Single Integer input.

Output Format:

Output displays Yes if condition satisfies else prints No.

Example Input:

14

Output:

Yes

Example Input:

13

Output:

No

Ex. No. : 4.9 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Product of single digit

Given a positive integer N, check whether it can be represented as a product of single digit numbers.

Program:

```
N = int(input())

temp = N

for divisor in range(2, 10):

    while temp % divisor == 0:

        temp //= divisor

if temp == 1:

    print("Yes")

else:

    print("No")
```

Input Format:

Single integer input.

Output Format:

Yes or No.

Example Input:

24

Output:

Yes

Example Input:

26

Output:

No

For example:

Input	**Result**
24	Yes

Ex. No. : 4.10 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Perfect Square After adding One

Given an integer N, check whether N the given number can be made a perfect square after adding 1 to it.

Program:

```
n=int(input())+1
a = int(n**0.5)
if(n==a*a):
   print("Yes")
else:
   print("No")
```

05 - List in Python

Sample Case 0
Sample Input 0
4
1
2
3
3
Sample Output 0
2

Explanation 0

· The sum of the first two elements, 1+2=3. The value of the last element is 3.

· Using zero based indexing, arr[2]=3 is the pivot between the two subarrays.

· The index of the pivot is 2.

Sample Case 1
Sample Input 1
3
1
2
1
Sample Output 1
1
Explanation 1

· The first and last elements are equal to 1.

· Using zero based indexing, arr[1]=2 is the pivot between the two subarrays.

· The index of the pivot is 1.

For example:

Input	Result
4 1 2 3 3	2
3 1 2 1	1

Ex. No. : 5.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Balanced Array

Given an array of numbers, find the index of the smallest array element (the pivot), for which the sums of all elements to the left and to the right are equal. The array may not be reordered.

Example
arr=[1,2,3,4,6]

- the sum of the first three elements, 1+2+3=6. The value of the last element is 6.
- Using zero based indexing, arr[3]=4 is the pivot between the two subarrays.
- The index of the pivot is 3.

Constraints

- $3 \leq n \leq 10^5$
- $1 \leq arr[i] \leq 2 \times 10^4$, where $0 \leq i < n$
- It is guaranteed that a solution always exists.

The first line contains an integer n, the size of the array arr.
Each of the next n lines contains an integer, arr[i], where $0 \leq i < n$.

Program:

```
n = int(input("Enter the size of the array: "))
arr = []
for i in range(n):
    element = int(input(f"Enter element {i+1} of {n}: "))
    arr.append(element)
total_sum = sum(arr)
left_sum = 0
pivot_index = -1
for i in range(n):
    total_sum -= arr[i]
    if left_sum == total_sum:
        pivot_index = i
        break

    left_sum += arr[i]
print(f"The index of the pivot is: {pivot_index}")
```

```
Input
1
3
1
3
5
4
Output:
1
Input
1
3
1
3
5
99
Output
0
```

For example:

Input	Result
1 3 1 3 5 4	1
1 3 1 3 5 99	0

Ex. No. : 5.2 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Check pair with difference k

Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[i] - A[j] = k, i != j.

Input Format

1. First line is number of test cases T. Following T lines contain:
2. N, followed by N integers of the array
3. The non-negative integer k

Output format

Print 1 if such a pair exists and 0 if it doesn’t.

Program:

```
T = int(input(""))
for _ in range(T):
    N = int(input(""))
    arr = []
    for _ in range(N):
        arr.append(int(input("")))
    k = int(input(""))
    arr_set = set(arr)
    found = 0
    for num in arr:
        if (num - k) in arr_set or (num + k) in arr_set:
            found = 1
            break
    print(found)
```

Sample Test Cases

Test Case 1

Input

```
7
23
45
23
56
45
23
40
```

Output

```
23 occurs 3 times
45 occurs 2 times
56 occurs 1 times
40 occurs 1 times
```

Ex. No.	**:** **5.3**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Count Elements

Complete the program to count frequency of each element of an array. Frequency of a particular element will be printed once.

Program:

```
n = int(input())
arr = []
for i in range(n):
    element = int(input())
    arr.append(element)
arr.sort()
processed = []
for i in range(n):
    if arr[i] not in processed:
        count = arr.count(arr[i])
        print(f"{arr[i]} occurs {count} times")
        processed.append(arr[i])
```

Example Input:

```
5
1
2
2
3
4
```

Output:

```
1 2 3 4
```

Example Input:

```
6
1
1
2
2
3
3
```

Output:

```
1 2 3
```

For example:

Input Result

```
5
1
2
2
3
4
1 2 3 4
6
1
1
2
2
3
3
1 2 3
```

Ex. No. : 5.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Distinct Elements in an Array

Program to print all the distinct elements in an array. Distinct elements are nothing but the unique (non-duplicate) elements present in the given array.

Input Format:

First line take an Integer input from stdin which is array length n.

Second line take n Integers which is inputs of array.

Output Format:

Print the Distinct Elements in Array in single line which is space Separated

Program:

```
n = int(input("Enter the number of elements in the array: "))

arr = []

for i in range(n):

    element = int(input(f"Enter element {i+1}: "))

    arr.append(element)

arr.sort()

prev_element = None

for element in arr:

    if element != prev_element:

        print(element, end=" ")

        prev_element = element
```

Sample Test Cases

Test Case 1

Input

```
1
3
4
5
6
7
8
9
10
11
2
```

Output

```
ITEM to be inserted:2
After insertion array is:
1
2
3
4
5
6
7
8
9
10
11
```

Test Case 2

Input

```
11
22
33
55
66
77
88
99
110
120
44
```

Output

```
ITEM to be inserted:44
After insertion array is:
11
22
33
44
55
66
77
88
99
110
120
```

Ex. No. : 5.5 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Element Insertion

Consider a program to insert an element / item in the sorted array. Complete the logic by filling up required code in editable section. Consider an array of size 10. The eleventh item is the data is to be inserted.

Program:

```
arr = [None] * 10
print()
for i in range(10):
    arr[i] = int(input())
item = int(input())
print(f"ITEM to be inserted: {item}")
position = 0
while position < len(arr) and arr[position] < item:
    position += 1
arr.append(None)
for i in range(len(arr) - 1, position, -1):
    arr[i] = arr[i-1]
arr[position] = item
print("After insertion array is:")
for element in arr:
    print(element)
```

Sample Case 0
Sample Input 0
10
3
Sample Output 0
5
Explanation 0
Factoring n = 10 results in {1, 2, 5, 10}. Return the p = 3rd factor, 5, as the answer.
Sample Case 1
Sample Input 1
10
5
Sample Output 1
0
Explanation 1
Factoring n = 10 results in {1, 2, 5, 10}. There are only 4 factors and p = 5, therefore 0 is returned as the answer.
Sample Case 2
Sample Input 2
1
1
Sample Output 2
1
Explanation 2
Factoring n = 1 results in {1}. The p = 1st factor of 1 is returned as the answer.

For example:

Input	Result
10 3	5
10 5	0
1 1	1

Ex. No. : 5.6 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Find the Factor

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the p^{th} element of the list, sorted ascending. If there is no p^{th} element, return 0.

Constraints

$1 \leq n \leq 10^{15}$

$1 \leq p \leq 10^{9}$

The first line contains an integer n, the number to factor.

The second line contains an integer p, the 1-based index of the factor to return.

Program:

```
n = int(input())
p = int(input())
factors = []
for i in range(1, int(n**0.5) + 1):
    if n % i == 0:
        factors.append(i)
        if i != n // i:
            factors.append(n // i)
factors.sort()
if p <= len(factors):
    print(factors[p-1])
else:
    print(0)
```

Sample test case

Sample input

```
2
2
1
3
5
7
2
4
6
8
```

Sample Output

```
[[1, 3, 2, 4], [5, 7, 6, 8]]
```

Ex. No. : 5.7 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Merge List

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

Input:
m : row size
n: column size
list1 and list 2 : Two lists
Output
Zipped List : List which combined both list1 and list2

Program:

```
m = int(input())
n = int(input())
list1 = []
list2 = []
for _ in range(m):
   row = [int(input()) for _ in range(n)]
   list1.append(row)
for _ in range(m):
   row = [int(input()) for _ in range(n)]
   list2.append(row)
zipped_list = [list1[i] + list2[i] for i in range(m)]
print("Zipped List:", zipped_list)
```

Sample Input 1

```
5
1
2
3
6
9
4
2
4
5
10
```

Sample Output 1

```
1 2 3 4 5 6 9 10
```

Ex. No. : 5.8 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Merge Two Sorted Arrays Without Duplication

Output is a merged array without duplicates.

Input Format
N1 - no of elements in array 1
Array elements for array 1
N2 - no of elements in array 2
Array elements for array2
Output Format
Display the merged array

Program:

```
N1 = int(input())

array1 = [int(input()) for _ in range(N1)]

N2 = int(input())

array2 = [int(input()) for _ in range(N2)]

merged_array = sorted(set(array1 + array2))

print(" ".join(map(str, merged_array)))
```

For example, if there are 4 elements in the array:

```
5
6
5
7
```

If the element to search is 5 then the output will be:

```
5 is present at location 1
5 is present at location 3
5 is present 2 times in the array.
```

Sample Test Cases

Test Case 1

Input

```
4
5
6
5
7
5
```

Output

```
5 is present at location 1.
5 is present at location 3.
5 is present 2 times in the array.
```

Test Case 2

Input

```
5
67
80
45
97
100
50
```

Output

```
50 is not present in the array.
```

Ex. No. : 5.9 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Print Element Location

Write a program to print all the locations at which a particular element (taken as input) is found in a list and also print the total number of times it occurs in the list. The location starts from 1.

Program:

```
n = int(input())
print()
lst = [int(input()) for _ in range(n)]
target = int(input())
count = 0
location = 1
for element in lst:
    if element == target:
        print(f"Element found at location: {location}")
        count += 1
    location += 1
if count > 0:
    print(f"Total occurrences of the element: {count}")
else:
    print(f"{target} is not present in the array.")
```

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

Ex. No. : 5.10 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Strictly increasing

Write a Python program to check if a given list is strictly increasing or not. Moreover, If removing only one element from the list results in a strictly increasing list, we still consider the list true
Input:
n : Number of elements
List1: List of values
Output
Print "True" if list is strictly increasing or decreasing else print "False"

Program:

```
n = int(input())
list1 = [int(input()) for i in range(n)]
is_strictly_increasing = True
can_be_strictly_increasing = False
violations = 0
last_violation_index = -1
for i in range(n - 1):
    if list1[i] >= list1[i + 1]:
        violations += 1
        last_violation_index = i
        is_strictly_increasing = False

    if violations > 1:
        break
if not is_strictly_increasing and violations == 1:
    if last_violation_index == 0 or last_violation_index == n - 2:
        can_be_strictly_increasing = True
    elif list1[last_violation_index - 1] < list1[last_violation_index + 1] or \
         list1[last_violation_index] < list1[last_violation_index + 2]:
        can_be_strictly_increasing = True
if is_strictly_increasing or can_be_strictly_increasing:
    print("True")
else:
    print("False")
```

06 - Strings in Python

For example:

Input Result
rec@123
3
3
1

Ex. No. : 6.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Count Chars

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

Program:

```
input_string = input()

letters = digits = special_symbols = 0

for char in input_string:

   if char.isalpha():

      letters += 1

   elif char.isdigit():

      digits += 1

   else:

      special_symbols += 1


print(letters)

print(digits)

print(special_symbols)
```

Sample Input 1
a2b4c6

Sample Output 1
aabbbbcccccc

Ex. No.	**:** **6.2**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Decompress the String

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

Program:

```
s = input()

s_list = list(s)

i = 0

while i < len(s_list):

    if s_list[i].isdigit():

        num_str = s_list[i]

        i += 1

        while i < len(s_list) and s_list[i].isdigit():

            num_str += s_list[i]

            i += 1

        repeat_times = int(num_str) - 1

        s_list[i-len(num_str):i] = [s_list[i-len(num_str)-1]] * repeat_times

    else:

        i += 1

decompressed_string = ''.join(s_list)

print(decompressed_string)
```

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.

Ex. No. : 6.3 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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.

Program:

```
S1 = input()

S2 = input()

N = int(input())

result = ''

for char in S1:

    if char in S2 and char not in result:

        result += char

        if len(result) == N:

            break

print(result)
```

Sample Input 1
experience
enc

Sample Output 1
xpri

Ex. No. : 6.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Remove Characters

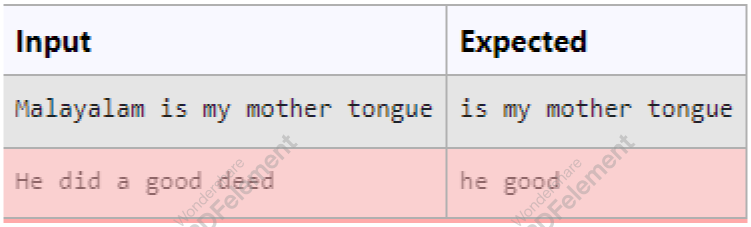
Given two Strings s1 and s2, remove all the characters from s1 which is present in s2.

Constraints
1<= string length <= 200

Program:

```
s1 = input()

s2 = input()

result = ''

for char in s1:

    if char not in s2:

        result += char

print(result)
```

For example:



Input	Expected
Malayalam is my mother tongue	is my mother tongue
He did a good deed	he good

Ex. No.	**:** **6.5**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

<u>Remove Palindrome Words</u>

String should contain only the words are not palindrome.

Sample Input 1
Malayalam is my mother tongue

Sample Output 1
is my mother tongue

Program:

```
s = input()
words = s.split()
non_palindromes = []
for word in words:
    if word.lower() != word.lower()[::-1]:
        non_palindromes.append(word)
result = ' '.join(non_palindromes)
print(result)
```

For example:

Input Result
Wipro Technologies Bangalore
TECHNOLOGIES
Hello World
WORLD
Hello
LESS

Ex. No. : 6.6 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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.

Program:

```
s = input()

words = s.split()

result = words[1].upper() if len(words) >= 2 else "LESS"

print(result)
```

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#

Ex. No. : 6.7 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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.

Program:

```
s = input()

alphabets = [c for c in s if c.isalpha()]

reversed_alphabets = alphabets[::-1]

result_list = []

j = 0

for i in range(len(s)):

    if s[i].isalpha():

        result_list.append(reversed_alphabets[j])

        j += 1

    else:

        result_list.append(s[i])

result = ''.join(result_list)

print(result)
```

For example:

Input Result
Yn
PYnative
True

Ex. No. :	**6.8**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

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".

Program:

```
s1 = input()
s2 = input()
result = True
for char in s1:
    if char not in s2:
        result = False
        break
print("True" if result else "False")
```

Input:
first
second
first
third
second

then your program should display:

Output:
first
second
third

Ex. No. : 6.9 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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:

Program:

```
items=[]
while True:
    try:
        a=input()
        if not a:
            break
        if a not in items:
            items.append(a)
    except EOFError as e:
        break
for i in items:
    print(i)
```

Example Input/Output 1:

Input:

```
vijayakumar.r@rajalakshmi.edu.in
```

Output:

```
edu.in
rajalakshmi
vijayakumar.r
```

Ex. No. : 6.10 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Username Domain Extension

Given a string S which is of the format 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

Program:

```
a = input()

b = "".join(a.split("@")[1:])

print(b[b.find(".")+1:])


print(b[:b.find(".")])


print(a.split("@")[0])
```

07 - Functions

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

Ex. No. : 7.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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

Program:

```
def abundant(n):

    div_sum = sum([divisor for divisor in range(1,n) if n % divisor ==0 ])

    if div_sum > n:

        return 'Yes'

    else:

        return 'No'
```

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

Ex. No. : 7.2 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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”.

Program:

```
def automorphic(inp):
    sq=inp*inp
    last=sq%(10**len(str(inp))
    if inp==last:
        return "Automorphic"
    else:
        return "Not Automorphic"
```

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

Ex. No. : 7.3 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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.

Program:

```
def productDigits(n):
    num = str(n)
    even = 1
    odd = 0
    for i,digit in enumerate(num):
        digit = int(digit)
        if(i+1)%2 == 0:
            even *= digit
        else:
            odd += digit
    return even % odd == 0
```

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

Ex. No. : 7.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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< $10e^{100000}$

Program:

```
def is_prime_digit(digit):
    return digit in [2, 3, 5, 7]


def christmasDiscount(n):
    discount = 0
    prime_digits = [2, 3, 5, 7]

    for digit in str(n):
        digit = int(digit)
        if is_prime_digit(digit):
            discount += digit

    return discount
```

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

Ex. No. **:** **7.5** **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

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

Program:

```
def coinChange(n):
    coin = [1,2,3,4]
    dp = [float('inf')]*(n+1)
    dp[0] = 0
    for i in range(1,n+1):
        for c in coin:
            if c <= i:
                dp[i]=min(dp[i],dp[i-c]+1)
    return dp[n]
```

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

Ex. No. :	**7.6**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Difference Sum

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

Program:

```
def digit_difference(num_str):
    even_sum = 0
    odd_sum = 0

    for i, digit in enumerate(num_str):
        if i % 2 == 0:
            even_sum += int(digit)
        else:
            odd_sum += int(digit)

    return abs(even_sum - odd_sum)
```

For example:

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

Ex. No. : 7.7 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Ugly number

A number is considered to be ugly if its only prime factors are 2, 3 or 5.
[1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, …] is the sequence of ugly numbers.
Task:
complete the function which takes a number n as input and checks if it's an ugly number. return ugly if it is ugly, else return not ugly
Hint:
An ugly number U can be expressed as: U = 2^a * 3^b * 5^c, where a, b and c are nonnegative integers.

Program:

```
def checkUgly(n):
    if n==1:
        return 'ugly'
    if n==0:
        return 'not ugly'
    if ( n % 2 == 0 ):
        return checkUgly(n // 2)
    if ( n % 3 == 0 ):
        return checkUgly(n // 3)
    if ( n % 5 == 0 ):
        return checkUgly(n // 5)
    return 'not ugly'
```

08 – Tuple/Set

Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

For example:

Input	**Result**
01010101010	Yes
010101 10101	No

Ex. No. : 8.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Binary String

Coders here is a simple task for you, Given string str. Your task is to check whether it is a binary string or not by using python set.

Program:

```
def is_binary_string(s):
    s = set(s)
    l = sorted(s)
    if( len(l)==2 and l[0]==’0’ and l[1]==’1’):
        return “Yes”
    else:
        return “No”
```

Examples:

Input: t = (5, 6, 5, 7, 7, 8), K = 13
Output: 2
Explanation:
Pairs with sum K(= 13) are {(5, 8), (6, 7), (6, 7)}.
Therefore, distinct pairs with sum K(= 13) are { (5, 8), (6, 7) }.
Therefore, the required output is 2.

For example:

Input	Result
1,2,1,2,5 3	1
1,2 0	0

Ex. No.	**:** 8.2	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Check Pair

Given a tuple and a positive integer k, the task is to find the count of distinct pairs in the tuple whose sum is equal to **K**.

Program:

```
t = tuple(map(int,input().split(',')))
inp = int(input())
s=set(t)
count = 0
for x in s:
    if inp - x in s:
        count += 1
res = count // 2
print(res)
```

Example 1:

Input: s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"
Output: ["AAAAACCCCC","CCCCCAAAAA"]

Example 2:

Input: s = "AAAAAAAAAAAAA"
Output: ["AAAAAAAAAA"]

For example:

Input	Result
AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA

Ex. No. : 8.3 Date:

Register No.: 230701289 Name: sanjay kumar.k

DNA Sequence

The **DNA sequence** is composed of a series of nucleotides abbreviated as 'A', 'C', 'G', and 'T'.

For example, "ACGAATTCCG" is a **DNA sequence**.

When studying **DNA**, it is useful to identify repeated sequences within the DNA.

Given a string s that represents a **DNA sequence**, return all the **10-letter-long** sequences (substrings) that occur more than once in a DNA molecule. You may return the answer in **any order**.

Program:

```
s = input()

s_count = {}

for i in range (len(s)-9):

    substring = s[i:i+10]

    s_count[substring] = s_count.get(substring,0)+1

rep_string = [substring for substring,count in s_count.items() if count > 1]

for i in rep_string:

    print(i)
```

Example 1:

Input: nums = [1,3,4,2,2]
Output: 2

Example 2:

Input: nums = [3,1,3,4,2]
Output: 3

For example:

Input	Result
1 3 4 4 2	4

Ex. No. : 8.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Print repeated no

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.There is only **one repeated number** in nums, return *this repeated number*. Solve the problem using set.

Program:

```
n = list(map(int,input().split()))
n_set=set()
for num in n:
    if num in n_set:
        print(num)
        break
    else:
        n_set.add(num)
```

Sample Input:

5 4

1 2 8 6 5

2 6 8 10

Sample Output:

1 5 10

3

Sample Input:

5 5

1 2 3 4 5

1 2 3 4 5

Sample Output:

NO SUCH ELEMENTS

For example:

Input	Result

5 4
1 2 8 6 5
2 6 8 10
	1 5 10
	3

Ex. No. : 8.5 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Remove repeated

Write a program to eliminate the common elements in the given 2 arrays and print only the non-repeating elements and the total number of such non-repeating elements.

Input Format:

The first line contains space-separated values, denoting the size of the two arrays in integer format respectively.

The next two lines contain the space-separated integer arrays to be compared.

Program:

```
def remove_common(a,b):
    common = list(set(a) - set(b))
    common1 = list(set(b) - set(a))
    common3 = common+common1
    if len(common3) > 0:
        for i in range(len(common3)):
            print(common3[i],'',end='')
        print('')
        print(len(common3))
    else:
        print('NO SUCH ELEMENTS')
num = input()
a = set (map(int,input().split()))
b = set (map(int,input().split()))
remove_common(a,b)
```

Example 1:

Input: text = "hello world", brokenLetters = "ad"

Output:

1

Explanation: We cannot type "world" because the 'd' key is broken.

For example:

Input	Result
hello world ad	1

Ex. No. : 8.6 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

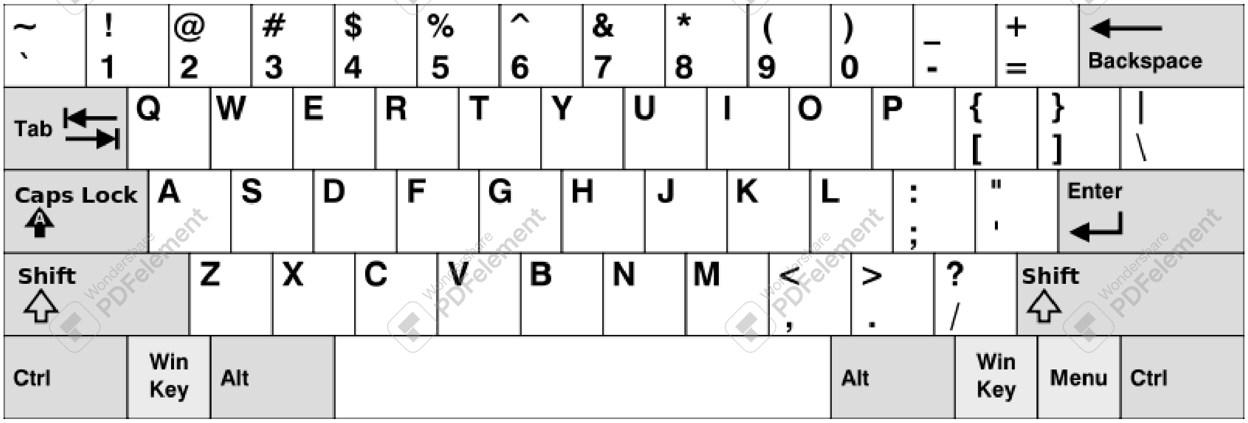
Malfunctioning Keyboard

There is a malfunctioning keyboard where some letter keys do not work. All other keys on the keyboard work properly.

Given a string text of words separated by a single space (no leading or trailing spaces) and a string brokenLetters of all distinct letter keys that are broken, return the number of words in text you can fully type using this keyboard.

Program:

```
text = input()

text1 = text.lower()

bro_text = input()

words = text1.split()

validword = 0

for word in words:

    if any(letter in bro_text for letter in word):

        continue

    else:

        validword+=1

print(validword)
```


~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = Backspace
Tab Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " ' Enter
Shift Z X C V B N M < , > . ? / Shift
Ctrl Win Key Alt Alt Win Key Menu Ctrl

Example 1:

Input: words = ["Hello","Alaska","Dad","Peace"]
Output: ["Alaska","Dad"]

Example 2:

Input: words = ["omk"]
Output: []

Example 3:

Input: words = ["adsdf","sfd"]
Output: ["adsdf","sfd"]

For example:

Input	Result
4 Hello Alaska Dad Peace	Alaska Dad

Ex. No. : 8.7 Date:

Register No.: 230701289 Name: sanjay kumar.k

American keyboard

Given an array of strings words, return *the words that can be typed using letters of the alphabet on only one row of American keyboard like the image below.*

In the **American keyboard**:

- the first row consists of the characters "qwertyuiop",
- the second row consists of the characters "asdfghjkl", and
- the third row consists of the characters "zxcvbnm".

Program:

```
row1 = set('qwertyuiop')

row2 = set('asdfghjkl')

row3 = set('zxcvbnm')

num_words = int(input())

found = False

for _ in range(num_words):

    word = input()

    word_lower = word.lower()

    if all(char in row1 for char in word_lower) or \

       all(char in row2 for char in word_lower) or \

       all(char in row3 for char in word_lower):

        print(word)

        found = True

if not found:

    print("No words")
```

09 – Dictionary

Example 1:

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

Output: ["sweet","sour"]

Example 2:

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

Output: ["banana"]

Constraints:

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

s1 and s2 consist of lowercase English letters and spaces.

s1 and s2 do not have leading or trailing spaces.

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

Note:

Use dictionary to solve the problem

For example:

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

Ex. No. : 9.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Uncommon words

A sentence is a string of single-space separated words where each word consists only of lowercase letters.A word is uncommon if it appears exactly once in one of the sentences, and does not appear in the other sentence.

Given two sentences s1 and s2, return a list of all the uncommon words. You may return the answer in any order.

Program:

```
s1 = input().strip()
s2 = input().strip()
word1 = s1.split()
word2 = s2.split()
word_count1 = {}
word_count2 = {}
for word in word1:
    word_count1[word] = word_count1.get(word,0) + 1
for word in word2:
    word_count2[word] = word_count2.get(word,0) + 1
uncommon_word = set()
for word,count in word_count1.items():
    if count == 1 and word not in word_count2:
        uncommon_word.add(word)
for word,count in word_count2.items():
    if count == 1 and word not in word_count1:
        uncommon_word.add(word)
if len(uncommon_word) == 0:
    print('No uncommon words')
else:
    x=list(uncommon_word)
    print(*x)
```

Input : test_dict = {‘Gfg’ : [6, 7, 4], ‘best’ : [7, 6, 5]}

Output : {‘Gfg’: 17, ‘best’: 18}

Explanation : Sorted by sum, and replaced.

Input : test_dict = {‘Gfg’ : [8,8], ‘best’ : [5,5]}

Output : {‘best’: 10, ‘Gfg’: 16}

Explanation : Sorted by sum, and replaced.

Sample Input:

2

Gfg 6 7 4

Best 7 6 5

Sample Output

Gfg 17

Best 18

For example:

Input	Result
2 Gfg 6 7 4 Best 7 6 5	Gfg 17 Best 18

Ex. No. :	**9.2**	**Date:**	
Register No.:	230701289	**Name:**	sanjay kumar.k

Sort Dictionary by Values Summation

Give a dictionary with value lists, sort the keys by summation of values in value list.

Program:

```
try:
    T = int(input())
    result = {}
    for _ in range(T):
        key,*values = input().split()
        values = list(map(int,values))
        sum_value=sum(values)
        result[key]=sum_value
    sorted_results = dict(sorted(result.items(),key = lambda item: item[1]))
    for key, value in sorted_results.items():
        print(key,value)
except:
    print("No input provided")
```

Examples:

```
Input :  votes[] = {"john", "johnny", "jackie",
                "johnny", "john", "jackie",
                "jamie", "jamie", "john",
                "johnny", "jamie", "johnny",
                "john"};
Output : John
```

We have four Candidates with name as 'John', 'Johnny', 'jamie', 'jackie'. The candidates John and Johny get maximum votes. Since John is alphabetically smaller, we print it. Use dictionary to solve the above problem

Sample Input:

10
John
John
Johny
Jamie
Jamie
Johny
Jack
Johny
Johny
Jackie

Sample Output:

Johny

For example:

Input	Result
10 John John Johny Jamie Jamie Johny Jack Johny Johny Jackie	Johny

Ex. No. : 9.3 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Winner of Election

Given an array of names of candidates in an election. A candidate name in the array represents a vote cast to the candidate. Print the name of candidates received Max vote. If there is tie, print a lexicographically smaller name.

Program:

```
try:
    n=int(input())
    votes = {}
    for _ in range(n):
        candidate = input()
        if candidate in votes:
            votes[candidate] += 1
        else:
            votes[candidate] = 1
    max_votes = max(votes.values())
    winner = [candidate for candidate, votes in votes.items() if votes == max_votes]
    winner = min(winner)
    print(winner)
except EOFError:
    print("No input provided")
```

Sample input:

```
4
James 67 89 56
Lalith 89 45 45
Ram 89 89 89
Sita 70 70 70
```

Sample Output:

```
Ram
James Ram
Lalith
Lalith
```

Ex. No. : 9.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Student Record

Create a student dictionary for n students with the student name as key and their test mark assignment mark and lab mark as values. Do the following computations and display the result.
1.Identify the student with the highest average score
2.Identify the student who as the highest Assignment marks
3.Identify the student with the Lowest lab marks
4.Identify the student with the lowest average score
Note:
If more than one student has the same score display all the student names

Program:

```
from operator import itemgetter

num = int(input())

students = []

for i in range(num):

    name, test, assgn, lab = input().split(" ")

    test, assgn, lab = int(test), int(assgn), int(lab)

    students.append({"name": name, "test": test, "assgn": assgn, "lab": lab, "avg" : (test + assgn + lab) / 3})

def get_names(items, field, comp):

    return sorted([i["name"] for i in items if i[field] == comp(items, key=itemgetter(field))[field]])

print(*get_names(students, "avg", max))

print(*get_names(students ,"assgn", max))

print(*get_names(students, "lab", min))

print(*get_names(students, "avg", min))
```

The points associated with each letter are shown below:

Points	Letters
1	A, E, I, L, N, O, R, S, T and U
2	D and G
3	B, C, M and P
4	F, H, V, W and Y
5	K
8	J and X
10	Q and Z

Sample Input

```
REC
```

Sample Output

```
REC is worth 5 points.
```

Ex. No.	**:**	**9.5**	**Date:**	
Register No.:		230701289	**Name:**	sanjay kumar.k

Scramble Score

In the game of Scrabble™, each letter has points associated with it. The total score of a word is the sum of the scores of its letters. More common letters are worth fewer points while less common letters are worth more points.
Write a program that computes and displays the Scrabble™ score for a word. Create a dictionary that maps from letters to point values. Then use the dictionary to compute the score.
A Scrabble™ board includes some squares that multiply the value of a letter or the value of an entire word. We will ignore these squares in this exercise.

Program:

```
letter_values = {
    'A':1,'E':1,'I':1,'L':1,'N':1,'O':1,'R':1,'S':1,'T':1,'U':1,
    'D':2,'G':2,
    'B':3,'C':3,'M':3,'P':3,
    'F':4,'H':4,'V':4,'W':4,'Y':4,
    'K':5,
    'J':8,'K':8,
    'Q':10,'Z':10,
}
word = input()
score = sum(letter_values.get(letter.upper(),0) for letter in word)
print(f"{word} is worth {score} points.")
```

10 - Searching & Sorting

For example:

Input	Result
5 6 5 4 3 8	3 4 5 6 8

Ex. No. : 10.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Merge Sort

Write a Python program to sort a list of elements using the merge sort algorithm.

Program:

```
def mergeSort(myList):
    if len(myList) > 1:
        mid = len(myList) // 2
        left = myList[:mid]
        right = myList[mid:]
        mergeSort(left)
        mergeSort(right)
        i=0
        j=0
        k=0
        while i < len(left) and j< len(right):
            if left[i] <= right[j]:
                myList[k]=left[i]
                i=i+1
            else:
                myList[k]=right[j]
                j+=1
            k+=1
        while i < len(left):
            myList[k]=left[i]
            i+=1
            k+=1
        while j < len(right):
            myList[k]=right[j]
            j+=1
            k+=1
n=int(input())
input_string=input()
list=input_string.split()
for i in range(len(list)):
    list[i]=int(list[i])
#print(list)
mergeSort(list)
for i in range(len(list)):
    print(list[i],end='')
    print(' ',end='')
```

Input Format

The first line contains an integer,n , the size of the list a .
The second line contains n, space-separated integers a[i].

Constraints

- 2<=n<=600
- 1<=a[i]<=2x10^{6}.

Output Format

You must print the following three lines of output:

1. List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list.
3. Last Element: lastElement, the *last* element in the sorted list.

Sample Input 0

```
3
1 2 3
```

Sample Output 0

```
List is sorted in 0 swaps.
First Element: 1
Last Element: 3
```

For example:

Input	Result
3 3 2 1	List is sorted in 3 swaps. First Element: 1 Last Element: 3
5 1 9 2 8 4	List is sorted in 4 swaps. First Element: 1 Last Element: 9

Ex. No. : 10.2 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Bubble Sort

Given an listof integers, sort the array in ascending order using the *Bubble Sort* algorithm above. Once sorted, print the following three lines:

1. List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list.
3. Last Element: lastElement, the *last* element in the sorted list.

For example, given a worst-case but small array to sort: a=[6,4,1]. It took 3 swaps to sort the array. Output would be

```
Array is sorted in 3 swaps.
First Element: 1
Last Element: 6
```

Program:

```
n=int(input())
ele=input()
ele1=ele.split()
for i in range(len(ele1)):
   ele1[i]=int(ele1[i])
s_count=0
for i in range(n-1):
   for j in range(0,n-i-1):
      if ele1[j]>ele1[j+1]:
         s_count=s_count+1
         ele1[j],ele1[j+1]=ele1[j+1],ele1[j]
print('List is sorted in',s_count,'swaps.')
print('First Element:',ele1[0])
print('Last Element:',ele1[n-1])
```

Input Format

The first line contains a single integer n , the length of A .
The second line contains n space-separated integers,A[i].

Output Format

Print peak numbers separated by space.

Sample Input

5

8 9 10 2 6

Sample Output

10 6

For example:

Input	Result
4 12 3 6 8	12 8

Ex. No. : 10.3 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Peak Element

Given an list, find peak element in it. A peak element is an element that is greater than its neighbors.

An element a[i] is a peak element if

A[i-1] <= A[i] >=a[i+1] for middle elements. [0<i<n-1]

A[i-1] <= A[i] for last element [i=n-1]

A[i]>=A[i+1] for first element [i=0]

Program:

```
n=int(input())

num=input()

a=[]

num1=num.split()

for i in range(len(num1)):

   num1[i]=int(num1[i])

if num1[0]>num1[1]:

   print(num1[0],end=")

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

   if num1[i]>num1[i-1] and num1[i]>num1[i+1]:

      print(",num1[i],end=")

if num1[n-1]>num1[n-2]:

   print(",num1[n-1])
```

For example:

Input	Result
1 2 3 5 8 6	False
3 5 9 45 42 42	True

Ex. No. : 10.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Binary Search

Write a Python program for binary search.

Program:

```
def binary_search(arr,low,high,x):
    if high >= low:
        mid=(low + high)
        if arr[mid]==x:
            return mid
        elif arr[mid]> x:
            return binary_search(arr,low,mid-1,x)
        else:
            return binary_search(arr,mid+1,high,x)
    else:
        return -1
input_string=input()
list=input_string.split(",")
for i in range(len(list)):
    list[i]=int(list[i])
x=int(input())
result=binary_search(list,0,len(list)-1,x)
if result!=-1:
    print('True')
else:
    print('False')
```

Input:

```
1 68 79 4 90 68 1 4 5
```

output:

```
 1 2
 4 2
 5 1
 68 2
 79 1
90 1
```

For example:

Input	Result

4 3 5 3 4 5	3 2
	4 2
	5 2

Ex. No. : 10.5 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Frequency of Elements

To find the frequency of numbers in a list and display in sorted order.

Constraints:

1<=n, arr[i]<=100

Program:

```
num = input()

r_num =[]

ran_num = num.split()

for i in range(len(ran_num)):

    ran_num[i]=int(ran_num[i])

ran_num.sort()

r_d=list(set(ran_num))

for i in range(len(r_d)):

    count=0

    for j in range(len(ran_num)):

        if r_d[i]==ran_num[j]:

            count=count+1

           #print("arvijayakumar")

    print(r_d[i],end='')

    print('',count)
```

Exception Handling

For example:

Input	Result
10 2	Division result: 5.0 Modulo result: 0
7 3	Division result: 2.3333333333333335 Modulo result: 1
8 0	Error: Cannot divide or modulo by zero.

Ex. No. : 11.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Division and Modulo Calculator

Write a Python program that performs division and modulo operations on two numbers provided by the user. Handle division by zero and non-numeric inputs.

Input Format:

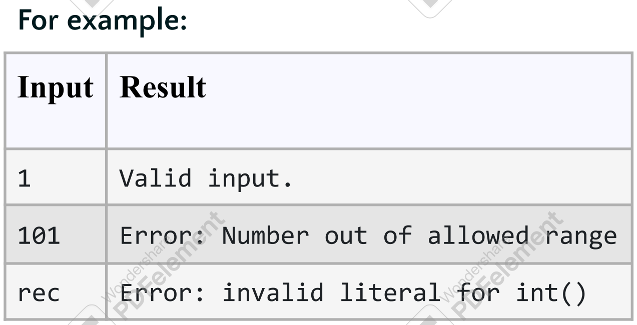
Two lines of input, each containing a number.

Output Format:

Print the result of division and modulo operation, or an error message if an exception occurs.

Program:

```
try:
        numerator_input = input().strip()
        numerator = float(numerator_input)
        denominator_input = input().strip()
        denominator = float(denominator_input)
        division_result = numerator / denominator
        print(f"Division result: {division_result}")
        modulo_result = numerator % denominator
        print(f"Modulo result: {modulo_result}")
except ValueError:
        print("Error: Non-numeric input provided.")
except ZeroDivisionError:
        print("Error: Cannot divide or modulo by zero.")
```



For example:

Input	Result
1	Valid input.
101	Error: Number out of allowed range
rec	Error: invalid literal for int()

Ex. No. : 11.2 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Integer Range Validator

Write a Python script that asks the user to enter a number within a specified range (e.g., 1 to 100). Handle exceptions for invalid inputs and out-of-range numbers.

Input Format:

User inputs a number.

Output Format:

Confirm the input or print an error message if it's invalid or out of range.

Program:

```
try:
    user_input = input("Please enter a number between 1 and 100: ").strip()
    number = int(user_input)
    if 1 <= number <= 100:
        print("Valid input.")
    else:
        print("Error: Number out of allowed range.")
except ValueError:
    print("Error: invalid literal for int()")
```

For example:

Input	Result

`10`
`2`
	`5.0`

`10`
`0`
	`Error: Cannot divide or modulo by zero.`

`ten`
`5`
	`Error: Non-numeric input provided.`

Ex. No. **:** **11.3** **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Robust Division Calculator

Develop a Python program that safely performs division between two numbers provided by the user. Handle exceptions like division by zero and non-numeric inputs.

Input Format: Two lines of input, each containing a number.

Output Format: Print the result of the division or an error message if an exception occurs.

Program:

```
try:
        numerator_input = input("Please enter the numerator: ").strip()
        numerator = float(numerator_input)
        denominator_input = input("Please enter the denominator: ").strip()
        denominator = float(denominator_input)
        result = numerator / denominator
        print(f"The result of the division is {result:.2f}")
except ValueError:
        print("Error: Non-numeric input provided.")
except ZeroDivisionError:
        print("Error: Cannot divide by zero.")
```

For example:

Input	Result
16	The square root of 16.0 is 4.00
-4	Error: Cannot calculate the square root of a negative number.
rec	Error: could not convert string to float

Ex. No. : 11.4 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Safe Square Root Calculator

Develop a Python program that safely calculates the square root of a number provided by the user. Handle exceptions for negative inputs and non-numeric inputs.

Input Format:

User inputs a number.

Output Format:

Print the square root of the number or an error message if an exception occurs.

Program:

```
import math
user_input = input("Please enter a number: ").strip()
try:
    number = float(user_input)
    if number < 0:
        print("Error: Cannot calculate the square root of a negative number.")
    else:
        square_root = math.sqrt(number)
        print(f"The square root of {number:.2f} is {square_root:.2f}")
except ValueError:
    print(f"Error: could not convert string to float")
```

For example:

Input	Result
twenty	Error: Please enter a valid age.
25	You are 25 years old.
-1	Error: Please enter a valid age.

Ex. No. : 11.5 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Validated User Input

Write a Python program that asks the user for their age and prints a message based on the age. Ensure that the program handles cases where the input is not a valid integer.

Input Format: A single line input representing the user's age.

Output Format: Print a message based on the age or an error if the input is invalid.

Program:

```
age_input = input("Please enter your age: ").strip()


try:
    age = int(age_input)
    if age < 0:
        print("Error: Please enter a valid age.")
    else:
        print(f"You are {age} years old.")
except ValueError:
    print("Error: Please enter a valid age.")
     #print("arvijayakumar")
```

Modules (Math/Collections)

Ex. No. : 12.1 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Power Of Four

Given an integer n, print *true if it is a power of four. Otherwise, print false*.

An integer n is a power of four, if there exists an integer x such that n == 4^x.

For example:

Input	Result
16	True
5	False

Program:

```
from math import log
n = int(input())

print(int(log(n, 4)) == log2(n, 4))
```

Ex. No. : 12.2 Date:

Register No.:230701289
Name: sanjay kumar.k

Count Pairs with Specific Absolute Difference

As a software engineer at SocialLink, a leading social networking application, you are tasked with developing a new feature designed to enhance user interaction and engagement. The company aims to introduce a system where users can form connections based on shared interests and activities. One of the feature's components involves analyzing pairs of users based on the activities they've participated in, specifically looking at the numerical difference in the number of activities each user has participated in.

Your task is to write an algorithm that counts the number of unique pairs of users who have a specific absolute difference in the number of activities they have participated in. This algorithm will serve as the backbone for a larger feature that recommends user connections based on shared participation patterns.

Problem Statement

Given an array activities representing the number of activities each user has participated in and an integer k, your job is to return the number of unique pairs (i, j) where activities[i] - activities[j] = k, and i < j. The absolute difference between the activities should be exactly k.

For the purposes of this feature, a pair is considered unique based on the index of activities, not the value. That is, if there are two users with the same number of activities, they are considered distinct entities.

Input Format

The first line contains an integer, n, the size of the array nums.
The second line contains n space-separated integers, nums[i].
The third line contains an integer, k.

Output Format

Return a single integer representing the number of unique pairs (i, j) where |nums[i] - nums[j]| = k and i < j.

Constraints:

$1 \leq n \leq 10^5$
$-10^4 \leq nums[i] \leq 10^4$
$0 \leq k \leq 10^4$

Sample Input 0

4
1 2 3 4
1

Sample Output 0
3

Explanation 0
There are three pairs with an absolute difference of 1: (1,2), (2,3), and (3,4).

Sample Input 1
5
1 3 1 5 4
0

Sample Output 1
1

Explanation 1
There is one pair with an absolute difference of 0: (1,1) considering the position in the array, not the value.

Sample Input 2
4
1 2 2 1
1

Sample Output 2
4

Explanation 2
The pairs with an absolute difference of 1 are:

- [**1**,**2**,2,1]
- [**1**,2,**2**,1]
- [1,**2**,2,**1**]
- [1,2,**2**,**1**]

Program:

```
def countKDiff (nums, k):
        count=0
        from collections import Counter
        mydict=Counter(nums)
        for i in mydict:
            if i+k in mydict:
                count=count+ mydict[i]*mydict[i+k]
        return count
```

Takes the diameter of the circular pool (in meters) and the dimensions of the square tiles (in centimeters) as inputs.

Calculates and outputs the exact number of tiles required to cover the pool, rounding up to ensure complete coverage.

For example:

Input	Result
`10 20`	`1964 tiles`
`10 30`	`873 tiles`

Ex. No. : 12.3 **Date:**

Register No.: 230701289 **Name:** sanjay kumar.k

Square Tiles

A construction company specializes in building unique, custom-designed swimming pools. One of their popular offerings is circular swimming pools. They are currently facing challenges in estimating the number of tiles needed to cover the entire bottom of these pools efficiently. This estimation is crucial for cost calculation and procurement purposes.

Problem Statement:

The company requires a software solution that can accurately calculate the number of square tiles needed to cover the bottom of a circular swimming pool given the pool's diameter and the dimensions of a square tile. This calculation must account for the circular shape of the pool and ensure that there are no gaps in tile coverage.

Program:

```
import math
def calculate_tiles_needed(pool_diameter_meters, tile_side_cm):
   pool_diameter_cm = pool_diameter_meters * 100
   pool_radius_cm = pool_diameter_cm / 2
   pool_area_sq_cm = math.pi * (pool_radius_cm ** 2)
   tile_area_sq_cm = tile_side_cm ** 2
   number_of_tiles = math.ceil(pool_area_sq_cm / tile_area_sq_cm)
   return number_of_tiles
pool_diameter_meters = int(input())
tile_side_cm = int(input())
tiles_needed = calculate_tiles_needed(pool_diameter_meters, tile_side_cm)

print(f"Number of tiles needed: {tiles_needed}")
```

Input Format:

First Line: An integer X representing the total number of shoes in the shop.

Second Line: A space-separated list of integers representing the shoe sizes in the shop.

Third Line: An integer N representing the number of customer requests.

Next N Lines: Each line contains a pair of space-separated values:

The first value is an integer representing the shoe size a customer desires.

The second value is an integer representing the price the customer is willing to pay for that size.

Output Format:

Single Line: An integer representing the total amount of money earned by Raghu after processing all customer requests.

Constraints:

1≤X≤1000 — Raghu's shop can hold between 1 and 1000 shoes.

Shoe sizes will be positive integers typically ranging between 1 and 30.

1≤N≤1000 — There can be up to 1000 customer requests in a single batch.

The price offered by customers will be a positive integer, typically ranging from $5 to $100 per shoe.

For example:

Input	**Result**

10
2 3 4 5 6 8 7 6 5 18
6
6 55
6 45
6 55
4 40
18 60
10 50
	200

5
5 5 5 5 5
5
5 10
5 10
5 10
5 10
5 10
	50

Ex. No. : 12.4 Date:

Total Revenue

Raghu owns a shoe shop with a varying inventory of shoe sizes. The shop caters to multiple customers who have specific size requirements and are willing to pay a designated amount for their desired shoe size. Raghu needs an efficient system to manage his inventory and calculate the total revenue generated from sales based on customer demands.

Problem Statement:

Develop a Python program that manages shoe inventory and processes sales transactions to determine the total revenue generated. The program should handle inputs of shoe sizes available in the shop, track the number of each size, and match these with customer purchase requests. Each transaction should only proceed if the desired shoe size is in stock, and the inventory should update accordingly after each sale.

Program:

```
from collections import Counter
def calculate_total_revenue(shoe_inventory, customer_requests):
    inventory_counter = Counter(shoe_inventory)
    total_revenue = 0
    for size, price in customer_requests:
        if inventory_counter[size] > 0:
            total_revenue += price
            inventory_counter[size] -= 1
    return total_revenue

X = int(input())
shoe_inventory = list(map(int, input().split()))
N = int(input())
customer_requests = []
for _ in range(N):
          size, price = map(int, input().split())
          customer_requests.append((size, price))
total_revenue = calculate_total_revenue(shoe_inventory, customer_requests)

print(total_revenue)
```

Constraints:

Book titles and genres are strings.

Book titles can vary in length but will not exceed 100 characters.

Genres will not exceed 50 characters.

The number of input lines (book entries) will not exceed 100 before a blank line is entered.

For example:

Input	Result

`Introduction to Programming, Programming`
`Advanced Calculus, Mathematics`
	`Programming: Introduction to Programming`
	`Mathematics: Advanced Calculus`

`Fictional Reality, Fiction`
`Another World, Fiction`
	`Fiction: Fictional Reality, Another World`

Ex. No.	**:**	**12.5**	**Date:**
Register No.:	230701289	**Name:**	sanjay kumar.k

Book Titles

Rose manages a personal library with a diverse collection of books. To streamline her library management, she needs a program that can categorize books based on their genres, making it easier to find and organize her collection.

Problem Statement:

Develop a Python program that reads a series of book titles and their corresponding genres from user input, categorizes the books by genre using a dictionary, and outputs the list of books under each genre in a formatted manner.

Input Format:

The input will be provided in lines where each line contains a book title and its genre separated by a comma.

Input terminates with a blank line.

Output Format:

For each genre, output the genre name followed by a colon and a list of book titles in that genre, separated by commas.

Program:

```
from collections import Counter, defaultdict
library = defaultdict(list)
while True:
        line = input().strip()
        if not line:
                break
        try:
                title, genre = map(str.strip, line.split(','))
        except ValueError:
                print("Invalid input format. Please enter as 'title,
genre'")
               #print("arvijayakumar")
                continue
        library[genre].append(title)
for genre, titles in library.items():

        print(f"{genre}: {', '.join(titles)}")
```