**RAJALAKSHMI ENGINEERING COLLEGE**

**RAJALAKSHMI NAGAR, THANDALAM – 602 105**



|  |
| --- |
| **CS23221**  **PYTHON PROGRAMMING LAB** |
| **Laboratory Observation Note Book** |

**INDEX**

Reg.No. : 230701004 Name: S.Abhinaya Lakshmi

Year: 2023- 2024 Branch: B.E. CSE Sec: A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.**  **No.** | **Date** | **Title** | **Page**  **No.** | **Teacher’s Signature /**  **Remarks** |
| **Introduction to python-Variables-Datatypes-Input/Output-Formatting** | | | | |
| 1.1 | 12.3.24 | Converting Input Strings |  |  |
| 1.2 | 12.3.24 | Gross salary |  |  |
| 1.3 | 12.3.24 | Square Root |  |  |
| 1.4 | 12.3.24 | Gain percent |  |  |
| 1.5 | 12.3.24 | Deposits |  |  |
| 1.6 | 12.3.24 | Carpenter |  |  |
| **Operators in Python** | | | | |
| 2.1 | 19.3.24 | Widgets and Gizmos |  |  |
| 2.2 | 19.3.24 | Doll Sings |  |  |
| 2.3 | 19.3.24 | Birthday party |  |  |
| 2.4 | 19.3.24 | Hamming Weight |  |  |
| 2.5 | 19.3.24 | Compound Interest |  |  |
| 2.6 | 19.3.24 | Eligible to donate blood |  |  |
| 2.7 | 19.3.24 | C or D |  |  |
| 2.8 | 19.3.24 | Troy Battle |  |  |
| 2.9 | 19.3.24 | Tax and Tip |  |  |
| 2.10 | 19.3.24 | Return last digit of the given number |  |  |
| **Selection Structures in Python** | | | | |
| 3.1 | 26.3.24 | Admission eligibility |  |  |
| 3.2 | 26.3.24 | Classifying triangles |  |  |
| 3.3 | 26.3.24 | Electricity Bill |  |  |
| 3.4 | 26.3.24 | IN/OUT |  |  |
| 3.5 | 26.3.24 | Vowel or Constant |  |  |
| 3.6 | 26.3.24 | Leap Year |  |  |
| 3.7 | 26.3.24 | Month name to Days |  |  |
| 3.8 | 26.3.24 | Pythagorean triple |  |  |
| 3.9 | 26.3.24 | Second Last Digit |  |  |
| 3.10 | 26.3.24 | Chinese Zodiac |  |  |
| **Algorithmic Approach: Iteration Control Structures** | | | | |
| 4.1 | 2.4.24 | Factors of a Number |  |  |
| 4.2 | 2.4.24 | Non-Repeated Digits Count |  |  |
| 4.3 | 2.4.24 | Prime Checking |  |  |
| 4.4 | 2.4.24 | Next Perfect Square |  |  |
| 4.5 | 2.4.24 | Nth Fibonacci |  |  |
| 4.6 | 2.4.24 | Disarium Number |  |  |
| 4.7 | 2.4.24 | Sum of Series |  |  |
| 4.8 | 2.4.24 | Unique Digits Count |  |  |
| 4.9 | 2.4.24 | Product of single digits |  |  |
| 4.10 | 2.4.24 | Perfect Square After adding One |  |  |
| **Strings in Python** | | | | |
| 5.1 | 16.4.24 | Count chars |  |  |
| 5.2 | 16.4.24 | Decompress the String |  |  |
| 5.3 | 16.4.24 | First N Common Characters |  |  |
| 5.4 | 16.4.24 | Remove Characters |  |  |
| 5.5 | 16.4.24 | Remove Palindrome Words |  |  |
| 5.6 | 16.4.24 | Return Second Word in Uppercase |  |  |
| 5.7 | 16.4.24 | Reverse String |  |  |
| 5.8 | 16.4.24 | String characters balance Test |  |  |
| 5.9 | 16.4.24 | Unique Names |  |  |
| 5.10 | 16.4.24 | Username Domain Extension |  |  |
| **List in Python** | | | | |
| 6.1 | 23.4.24 | Monotonic array |  |  |
| 6.2 | 23.4.24 | Check pair with difference k . |  |  |
| 6.3 | 23.4.24 | Count Elements |  |  |
| 6.4 | 23.4.24 | Distinct Elements in an Array |  |  |
| 6.5 | 23.4.24 | Element Insertion |  |  |
| 6.6 | 23.4.24 | Find the Factor |  |  |
| 6.7 | 23.4.24 | Merge list |  |  |
| 6.8 | 23.4.24 | Merge Two Sorted Arrays Without Duplication |  |  |
| 6.9 | 23.4.24 | Print Element Location |  |  |
| 6.10 | 23.4.24 | Strictly increasing |  |  |
| **Tuples & Set** | | | | |
| 7.1 | 21.5.24 | Distinct pairs |  |  |
| 7.2 | 21.5.24 | Check if binary |  |  |
| 7.4 | 21.5.24 | Print repeated no |  |  |
| 7.5 | 21.5.24 | Remove repeated |  |  |
| 7.7 | 21.5.24 | American keyboard |  |  |
| **Dictionary** | | | | |
| 8.1 | 28.5.24 | Uncommon Words |  |  |
| 8.2 | 28.5.24 | Sort Dictionary By Values Summation |  |  |
| 8.3 | 28.5.24 | Winner Of Election |  |  |
| 8.4 | 28.5.24 | Student Record |  |  |
| 8.5 | 28.5.24 | Scramble Score |  |  |
| **Functions** | | | | |
| 9.1 | 28.5.24 | Abundant Number |  |  |
| 9.2 | 28.5.24 | Automorphic number or not |  |  |
| 9.3 | 28.5.24 | Check Product of Digits |  |  |
| 9.4 | 28.5.24 | Christmas Discount |  |  |
| 9.7 | 28.5.24 | Ugly number |  |  |
| **Searching & Sorting** | | | | |
| 10.1 | 28.5.24 | Merge Sort |  |  |
| 10.2 | 28.5.24 | Bubble Sort |  |  |
| 10.3 | 28.5.24 | Peak Element |  |  |
| 10.4 | 28.5.24 | Binary Search |  |  |
| 10.5 | 28.5.24 | Frequency of Numbers |  |  |
|  |  | **Exceptions** |  |  |
| 11.1 | 4.6.24 | Check Age |  |  |
| 11.2 | 4.6.24 | Check age |  |  |
| 11.3 | 4.6.24 | Result of division |  |  |
| 11.4 | 4.6.24 | Result of division |  |  |
| 11.4 | 4.6.24 | Valid and Invalid input |  |  |
|  |  | **Modules** |  |  |
| 12.1 | 4.6.24 | Academic evaluation |  |  |
| 12.2 | 4.6.24 | Software enginner |  |  |
| 12.3 | 4.6.24 | Integer |  |  |
| 12.4 | 4.6.24 | Construction company |  |  |
| 12.5 | 4.6.24 | Library |  |  |

**01 - Introduction to Python-Variables-Datatypes**

**Input/Output-Formatting**

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

**Register No.: 230701004 Name:**

[**Converting Input Strings**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6373)

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

*Sample Input:*

10

10.9

*Sample Output:*

10, <class 'int'>

10.9, <class 'float'>

**For example:**

| **Input** | **Result** |
| --- | --- |
| 10  10.9 | 10,<class 'int'>  10.9,<class 'float'> |

x=int(input())

y=float(input())

print(x,",",type(x),sep="")

print(round(y,1),",",type(y),sep="")

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

**Register No.: 230701004 Name:**

[**Gross Salary**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6374)

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.

*Sample Input:*

10000

*Sample Output:*

16000

**For example:**

| **Input** | **Result** |
| --- | --- |
| 10000 | 16000 |

x=int(input())

y= (60/100)\*x

print(int(x+y))

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

**Register No.: 230701004 Name:**

**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.

Sample Input:

8.00

Sample Output:

2.828

**For example:**

| **Input** | **Result** |
| --- | --- |
| 14.00 | 3.742 |

import math

x=float(input())

y=math.sqrt(x)

print("{:.3f}".format(y))

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

**Register No.: 230701004 Name:**

[**Gain percent**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6376)

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.

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. |

x=int(input())

y=int(input())

z=int(input())

a=z-(x+y)

b= a/(x+y)

c= b\*100

print("{:.2f}".format(c),"is the gain percent.")

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

**Register No.: 230701004 Name:**

[**Deposits**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6378)

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.

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. |

x=int(input())

y=int(input())

a=(0.10\*x)

b=(0.25\*y)

z=a+b

print("Your total refund will be $","{:.2f}".format(z),".",sep="")

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

**Register No.: 230701004 Name:**

[**Carpenter**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6379)

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

**Sample Input:**

450

**Sample Output:**

weekdays 10.38

weekend 0.38

| **Input** | **Result** |
| --- | --- |
| 450 | weekdays 10.38  weekend 0.38 |

**For example:**

x=int(input())

n= (x-500)/130

z= abs(n)

y=z+10

print("weekdays","{:.2f}".format(y))

print("weekend","{:.2f}".format(z))

### [**02- Operators in Python**](https://www.rajalakshmicolleges.net/moodle/course/view.php?id=84#section-2)

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

**Register No.: 230701004 Name:**

[**Widgets and Gizmos**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5695)

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.

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. |

w=int(input())

g=int(input())

print("The total weight of all these widgets and gizmos is ",(w\*75)+(g\*112)," grams.",sep="")

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

**Register No.: 230701004 Name:**

[**Doll Sings**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5696)

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.

Sample Input

10

Sample Output

True

Explanation:

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

|  |  |
| --- | --- |
| **Input** | **Result** |
| 101 | False |

x=int(input())

if(x>0 and x<100):

print("True")

else:

print("False")

**Ex. No. : 2.3 Date:**

**Register No.: 230701004 Name:**

[**Birthday Party**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5697)

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.

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

For example:

| **Input** | **Result** |
| --- | --- |
| 5  25  23  20  10 | True False True True |

friends=int(input())

p1=int(input())

p2=int(input())

p3=int(input())

p4=int(input())

l=[p1,p2,p3,p4]

for i in l:

if (i%friends==0):

print("True ",end="")

else:

print("False ",end="")

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

**Register No.: 230701004 Name:**

[**Hamming Weight**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5698)

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.

**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.

**For example:**

|  |  |
| --- | --- |
| **Input** | **Result** |
| 3 | 2 |

x=int(input())

count=0

if (x>=0 and x<=15):

y=bin(x)

for i in y:

if(i=='1'):

count=count+1

print(count)

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

**Register No.: 230701004 Name:**

[**Compound Interest**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5699)

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.

**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

n=int(input())

print("Balance as of end of Year 1: $","{:.2f}".format(n+(.04\*n)),".",sep="")

num=(n+(.04\*n))

print("Balance as of end of Year 2: $","{:.2f}".format(num+(.04\*num)),".",sep="")

num1=(num+(.04\*num))

print("Balance as of end of Year 3: $","{:.2f}".format(num1+(.04\*num1)),".",sep="")

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

**Register No.: 230701004 Name:**

**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.

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

x=int(input())

y=int(input())

if(x>=18 and y>40):

print(True)

else:

print(False)

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

**Register No.: 230701004**

**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.

**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

x=int(input())

if(x==0):

print("C")

else:

print("D")

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

**Register No.: 230701004**

**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.

**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

x=int(input())

y=int(input())

if(x%3==0 and y%2==0):

print("True")

else:

print("False")

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

**Register No.: 230701004 Name:**

[**Tax and Tip**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6030)

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.

Sample Input

100

Sample Output

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

num=int(input())

print("The tax is ","{:.2f}".format((.05\*num))," and the tip is","{:.2f}".format((.18\*num)),", making the total ","{:.2f}".format((num+(.18\*num)+(.05\*num))),sep="")

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

**Register No.: 230701004 Name:**

[**Return last digit of the given number**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=6031)

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| 123 | 3 |

x=int(input())

y= abs(x)%10

print(y)

### [**03 - Selection Structures in Python**](https://www.rajalakshmicolleges.net/moodle/course/view.php?id=84#section-3)

**Ex. No. : 3.1 Date:**

**Register No.: 230701004 Name:**

[**Admission Eligibility**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5706)

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

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 |

Maths=int(input())

Physics=int(input())

Chemistry=int(input())

total= (Maths+Physics+Chemistry)

if (Maths >= 65 and Physics >= 55 and Chemistry >= 50 or total >=180):

print("The candidate is eligible")

else:

print("The candidate is not eligible")

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

**Register No.: 230701004 Name:**

[**Classifying Triangles**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5707)

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.

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 |

s1=int(input())

s2=int(input())

s3=int(input())

if s1==s2 and s2==s3:

print("That's a equilateral triangle")

elif (s1==s2 and s2!=s3) or (s1==s3 and s3!=s2):

print("That's a isosceles triangle")

else:

print("That's a scalene triangle")

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

**Register No.: 230701004 Name:**

[**Electricity Bill**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5708)

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/-

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 |

current=float(input())

amt=0.00

if current<200:

amt=1.20\*current

elif current>=200 and current<400:

amt=1.50\*current

elif current>=400 and current<600:

amt=1.80\*current

else:

amt=2.00\*current

if amt<100:

amt=100

if amt>400:

amt=amt\*(15/100)+amt

print("{:.2f}".format(amt))

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

**Register No.: 230701004 Name:**

[**IN/OUT**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5709)

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.

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 |

x=int(input())

y=int(input())

if (x/2<y):

print("IN")

elif(x%2==y or x%2>y):

print("IN")

else:

print("OUT")

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

**Register No.: 230701004**

[**Vowel or Consonant**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5710)

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.

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.

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

**For example:**

Q=input()

if (Q=='a'or Q=='e' or Q=='i' or Q=='o' or Q=='u'):

print("It's a vowel.")

elif(Q=='y'):

print("Sometimes it's a vowel... Sometimes it's a consonant.")

else:

print("It's a consonant.")

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

**Register No.: 230701004**

[**Leap Year**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5711)

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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5753) 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.

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.

year=int(input())

if year%400==0:

print(year,"is a leap year.")

elif year%100==0:

print(year,"is not a leap year.")

elif year%4==0:

print(year,"is a leap year.")

else:

print(year,"is not a leap year.")

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

**Register No.: 230701004 Name:**

[**Month name to days**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5712)

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.

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. |

month=input()

if month=="January":

print(month,"has 31 days in it.")

elif month=="February":

print(month,"has 28 or 29 days in it.")

elif month=="March":

print(month,"has 31 days in it.")

elif month=="April":

print(month,"has 30 days in it.")

elif month=="May":

print(month,"has 31 days in it.")

elif month=="June":

print(month,"has 30 days in it.")

elif month=="July":

print(month,"has 31 days in it.")

elif month=="August":

print(month,"has 30 days in it.")

elif month=="September":

print(month,"has 31 days in it.")

elif month=="October":

print(month,"has 30 days in it.")

elif month=="November":

print(month,"has 31 days in it.")

else:

print(month,"has 31 days in it.")

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

**Register No.: 230701004 Name:**

[**Pythagorean triple**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5713)

Three numbers form a Pythagorean triple if the [sum of squares](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5759) 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".

**Sample Input**

3

5

4

**Sample Output**

Yes

| **Input** | **Result** |
| --- | --- |
| 3  4  5 | Yes |

**For example:**

x=int(input())

y=int(input())

z=int(input())

if (x\*\*2==y\*\*2+z\*\*2) or (y\*\*2==x\*\*2+z\*\*2) or (z\*\*2==y\*\*2+x\*\*2):

print("yes")

else:

print("no")

**Ex. No. : 3.9 Date:**

**Register No.: 230701004 Name:**

[**Second last digit**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5714)

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.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 197 | 9 |

x=input()

if len(x)>1:

print(x[-2])

else:

print("-1")**Ex. No. : 3.10 Date:**

**Register No.: 230701004 Name:**

**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.

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.

year=int(input())

if (year-2000)%12==0:

print(year,"is the year of the Dragon.")

elif (year-2000)%12==1:

print(year,"is the year of the Snake.")

elif (year-2000)%12==2:

print(year,"is the year of the Horse.")

elif (year-2000)%12==3:

print(year,"is the year of the Sheep.")

elif (year-2000)%12==4:

print(year,"is the year of the Monkey.")

elif (year-2000)%12==5:

print(year,"is the year of the Rooster.")

elif (year-2000)%12==6:

print(year,"is the year of the Dog.")

elif (year-2000)%12==7:

print(year,"is the year of the Pig.")

elif (year-2000)%12==8:

print(year,"is the year of the Rat.")

elif (year-2000)%12==9:

print(year,"is the year of the Ox.")

elif (year-2000)%12==10:

print(year,"is the year of the Tiger.")

else:

print(year,"is the year of the Hare.")

[**04 - Iteration Control Structures**](https://www.rajalakshmicolleges.net/moodle/course/view.php?id=84#section-4)

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

**Register No.: 230701004 Name:**

[**Factors of a number**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5720)

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

| **Input** | **Result** |  |
| --- | --- | --- |
| 20 | 1 2 4 5 10 20 |  |

**For example:Ex. No. : 4.2 Date:**

n= int(input())

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

if n%x==0:

print(x,end= ’ ‘)

**Register No.: 230701004 Name:**

[**Non**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5717) **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.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 292 | 1 |
| 1015 | 2 |
| 108 | 3 |
| 22 | 0 |

num=input()

digitcount= {}

for digit in num:

if digit in digitcount:

digitcount[digit] += 1

else:

digitcount[digit] - 1

nrc= 0

for count in digitcount.values():

if count = 1:

nrc+= 1

print(nrc)

**Ex. No. : 4.3 Date:**

**Register No.: 230701004 Name:**

**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.

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 |

num = int(input())

flag = False

if num == 1:

print(1)

elif num > 1:

for i in range(2, num):

if (num % i) == 0:

flag = True

break

if flag:

print(1)

else:

print(2)

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

**Register No.: 230701004 Name:**

**Next Perfect Square**

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

Input Format:

Integer input from stdin.

Output Format:

Perfect square greater than N.

Example Input:

10

Output:

16

import math

from math import sqrt

n= int(input())+1

while int(sqrt(n)!=sqrt(n):

n=n+1

print(n)

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

**Register No.: 230701004 Name:**

**Nth Fibonacci**

Write a [program](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) to return the nth number in the fibonacci series. The value of N will be passed to the [program](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) as input.

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**

n = int(input())

fib = [0, 1]

if n == 1:

print(0)

elif n == 2:

print(1)

else:

for i in range(2, n):

fib.append(fib[-1] + fib[-2])

print(fib[-1])

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

**Register No.: 230701004 Name:**

**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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) to print number is Disarium or not.

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 |

def calculateLength(n):

length = 0

while n != 0:

length = length + 1

n = n // 10

return length

num = int(input())

rem = 0

sum = 0

len = calculateLength(num)

n = num

while num > 0:

rem = num % 10

sum = sum + int(rem \*\* len)

num = num // 10

len = len - 1

if sum == n:

print("Yes")

else:

print("No")

**Ex. No. : 4.7 Date:**

**Register No.: 230701004 Name:**

**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)

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 |

def sumofseries(n):

total=0

term=1

for i in range(n):

total+= term

term = term\*10+1

return total

n = int(input())

if n>=0:

seriessum = sumofseries(n)

print(seriessum)

else:

print(“invalid”)

**Ex. No. : 4.8 Date:**

**Register No.: 230701004 Name:**

**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'.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 292 | 2 |
| 1015 | 3 |

def count\_unique\_digits(N):  
 return len(set(str(N)))

number= int(input())  
if number>=0:

unique\_count= count\_unique\_digits(number)

print(unique\_count)

else:

print(“no unique”)

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

**Register No.: 230701004 Name:**

**Product of single digit**

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

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

def isproduct(N):  
 if N< 10:

return True

for i in range (2,10):

if N%i==0:

if N//i<10:

return True

return False

num= int(input())

if isproduct(num):

print(“Yes”)

else:

print(“No”)

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

**Register No.: 230701004 Name:**

**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.

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 |

import math

def is\_square(N):

square\_root= math.isqrt(N+1)

return square\_root+square\_root == N+1

num = int(input())

if is\_square(num):  
 print(“Yes”)

else:

print(“No”)

### [**06 - List in Python**](https://www.rajalakshmicolleges.net/moodle/course/view.php?id=84#section-5)

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

**Register No.: 230701004 Name:**

**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 ≤ n ≤ 105

·         1 ≤ arr[i] ≤ 2 × 104, where 0 ≤ 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 ≤ i < n.

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 |

def find\_pivot\_index(arr):

total\_sum = sum(arr)

left\_sum = 0

for i, num in enumerate(arr):

if left\_sum == total\_sum - left\_sum - num:

return i

left\_sum += num

# Main function

if \_\_name\_\_ == "\_\_main\_\_":

n = int(input().strip()) # Size of the array

arr = [int(input().strip()) for \_ in range(n)] # Array elements

pivot\_index = find\_pivot\_index(arr)

print(pivot\_index)

**Ex. No. : 6.2 Date:**

**Register No.: 230701004 Name:**

**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.

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 |

def pair\_exists\_with\_difference(arr, k):

num\_set = set(arr)

for num in arr:

if num - k in num\_set and num - k != num:

return 1

if num + k in num\_set and num + k != num:

return 1

return 0

def main():

T = int(input().strip())

for \_ in range(T):

N = int(input().strip())

arr = list(map(int, input().strip().split()))

k = int(input().strip())

print(pair\_exists\_with\_difference(arr, k))

if \_\_name\_\_ == "\_\_main\_\_":

main()

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

**Register No.: 230701004 Name:**

**Count Elements**

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

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

n= int(input().strip())

arr= [int(input().strip()) for \_ in range(n)]

frequency = {}

for num in arr:

if num in frequency:

frequency[num]+= 1

else:

frequency[num]=1

for key, value in frequency.items():

print(f"{key} occurs {value} times")

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

**Register No.: 230701004 Name:**

**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

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

def distinct\_elements(arr):

distinct\_set = set(arr)

distinct\_list = list(distinct\_set)

return distinct\_list

def main():

# Input length of the array

n = int(input())

# Input array elements

arr = []

for \_ in range(n):

arr.append(int(input()))

# Find distinct elements

distinct\_arr = distinct\_elements(arr)

# Print distinct elements

print(\*distinct\_arr)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Ex. No. : 6.5 Date:**

**Register No.: 230701004 Name:**

**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.

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

# Function to insert an element into a sorted array

def insert\_into\_sorted\_array(arr, item):

# Find the correct position to insert the item while maintaining sorted order

index = 0

for i in range(len(arr)):

if arr[i] < item:

index = i + 1

else:

break

# Shift elements to the right to make space for the new item

arr.insert(index, item)

return "ITEM to be inserted:" + str(item) + "\nAfter insertion array is:\n" + "\n".join(map(str, arr))

# Main function

if \_\_name\_\_ == "\_\_main\_\_":

arr = [int(input().strip()) for \_ in range(10)] # Input sorted array

item = int(input().strip()) # Item to insert

result = insert\_into\_sorted\_array(arr, item)

print(result)

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

**Register No.: 230701004 Name:**

**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 pth element of the list, sorted ascending. If there is no pth element, return 0.

**Constraints**

1 ≤ n ≤ 1015

1 ≤ p ≤ 109

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.

**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 isreturned 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 |

import math

def find\_factor(n, p):

factors = []

sqrt\_n = int(math.sqrt(n))

for i in range(1, sqrt\_n + 1):

if n % i == 0:

factors.append(i)

if i != n // i: # Avoid duplicate factor for perfect squares

factors.append(n // i)

factors.sort()

if p <= len(factors):

return factors[p - 1]

else:

return 0

# Main function

if \_\_name\_\_ == "\_\_main\_\_":

n = int(input().strip()) # Number to factor

p = int(input().strip()) # 1-based index of the factor to return

result = find\_factor(n, p)

print(result)

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

**Register No.: 230701004 Name:**

**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

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]]

def zip\_lists(list1, list2):

# Check if both lists have the same dimensions

if len(list1) != len(list2):

return "Error: Lists must have the same number of rows"

for i in range(len(list1)):

if len(list1[i]) != len(list2[i]):

return "Error: Lists must have the same number of columns in each row"

# Zipping the lists

zipped\_list = [list1[i] + list2[i] for i in range(len(list1))]

return zipped\_list

def get\_input(m, n):

list\_input = []

for i in range(m):

row = []

for j in range(n):

element = int(input())

row.append(element)

list\_input.append(row)

return list\_input

def main():

# Get input from user

m = int(input())

n = int(input())

list1 = get\_input(m, n)

list2 = get\_input(m, n)

# Zip the lists

result = zip\_lists(list1, list2)

# Print the zipped list

print(result)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Ex. No. : 6.8 Date:**

**Register No.: 230701004 Name:**

**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

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

def merge\_arrays(arr1, arr2):

merged\_array = arr1 + arr2

merged\_array = list(set(merged\_array)) # Remove duplicates

merged\_array.sort() # Sort the merged array

return merged\_array

if \_\_name\_\_ == "\_\_main\_\_":

# Input array 1

n1 = int(input().strip())

arr1 = [int(input().strip()) for \_ in range(n1)]

# Input array 2

n2 = int(input().strip())

arr2 = [int(input().strip()) for \_ in range(n2)]

# Merge arrays and remove duplicates

merged\_array = merge\_arrays(arr1, arr2)

# Print the merged array

print(" ".join(map(str, merged\_array)))

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

**Register No.: 230701004 Name:**

**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.

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.

def find\_element\_locations(lst, element):

locations = []

count = 0

for i, num in enumerate(lst, 1):

if num == element:

locations.append(i)

count += 1

return locations, count

# Input

n = int(input())

lst = []

for \_ in range(n):

lst.append(int(input()))

element\_to\_search = int(input())

# Finding locations and count

locations, count = find\_element\_locations(lst, element\_to\_search)

# Output

for location in locations:

print("{} is present at location {}.".format(element\_to\_search, location))

if count==0:

print("{} is not present in the array.".format(element\_to\_search))

else:

print("{} is present {} times in the array.".format(element\_to\_search, count))

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

**Register No.: 230701004 Name:**

**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"

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

def check\_increasing(lst):

# Check if the list is strictly increasing

increasing = all(lst[i] < lst[i+1] for i in range(len(lst)-1))

if increasing:

return True

# Check if removing one element results in a strictly increasing list

for i in range(len(lst)):

temp\_lst = lst[:i] + lst[i+1:]

if all(temp\_lst[j] < temp\_lst[j+1] for j in range(len(temp\_lst)-1)):

return True

return False

# Input

n = int(input())

List1 = list(map(int, input().split()))

# Check if the list is strictly increasing or not

result = check\_increasing(List1)

# Output

if result:

print("True")

**else:**

**print("False")**

### **05 - Strings in Python**

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

**Register No.: 230701004 Name:**

**Count Chars**

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

For example:

Input Result

rec@123

3

3

1

string=input()

alphabets=digits=special=0

for i in range(len(string)):

if(string[i].isalpha()):

alphabets=alphabets+1

elif(string[i].isdigit()):

digits=digits+1

else:

special=special+1

print(alphabets)

print(digits)

print(special)

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

**Register No.: 230701004 Name:**

**Decompress the String**

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

Sample Input 1

a2b4c6

Sample Output 1

aabbbbcccccc

a= list(input(“enter:”))

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

print(a[i]\*int(a[i+1]),end=” ”)

**Ex. No. : 5.3 Date:**

**Register No.: 230701004 Name:**

**First N Common Chars**

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

Input Format:

The first line contains S1.

The second line contains S2.

The third line contains N.

Output Format:

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

Boundary Conditions:

2 <= N <= 10

2 <= Length of S1, S2 <= 1000

Example Input/Output 1:

Input:

abcbde

cdefghbb

3

Output:

bcd

Note:

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

def common\_characters(S1, S2, N):

common\_chars = ""

count = 0

for char in S1:

if char in S2 and char not in common\_chars:

common\_chars += char

count += 1

if count == N:

break

return common\_chars

S1 = input().strip()

S2 = input().strip()

N = int(input().strip())

print(common\_characters(S1, S2, N))

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

**Register No.: 230701004 Name:**

**Remove Characters**

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

Constraints

1<= string length <= 200

Sample Input 1

experience

enc

Sample Output 1

xpri

def remove\_chars(S1, S2):

S2\_set = set(S2)

result = ""

for char in S1:

if char not in S2\_set:

result += char

return result

S1 = input().strip()

S2 = input().strip()

result = remove\_chars(S1, S2)

print(result) # Output: "he"

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

**Register No.: 230701004 Name:**

**Remove Palindrome Words**

String should contain only the words are not palindrome.

Sample Input 1

Malayalam is my mother tongue

Sample Output 1

is my mother tongue

**For example:**



def is\_palindrome(word):

return word == word[::-1]

def remove\_palindrome\_words(sentence):

words = sentence.lower().split()

non\_palindrome\_words = [word for word in words if not is\_palindrome(word)]

return ' '.join(non\_palindrome\_words)

input\_sentence = input().strip()

result = remove\_palindrome\_words(input\_sentence)

print(result)

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

**Register No.: 230701004 Name:**

**Return Second World in Uppercase**

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

For example:

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

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

If input is “Hello” the program should return “LESS”

NOTE 1: If input is a sentence with less than 2 words, the program should return the word “LESS”.

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

For example:

Input Result

Wipro Technologies Bangalore

TECHNOLOGIES

Hello World

WORLD

Hello

LESS

def get\_second\_word\_uppercase(sentence):

words = sentence.split()

if len(words) >= 2:

return words[1].upper()

else:

return "LESS"

# Example usage:

input\_sentence = input().strip()

result = get\_second\_word\_uppercase(input\_sentence)

print(result)

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

**Register No.: 230701004 Name:**

**Revers String**

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

Input:

A&B

Output:

B&A

Explanation: As we ignore '&' and

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

For example:

|  |  |
| --- | --- |
| **Input** | **Result** |
| A&x# | X&A# |

def reverseString(text):

index = -1

for i in range(len(text)-1, int((len(text)/2)-1), -1):

if text[i].isalpha():

temp = text[i]

while True:

index += 1

if text[index].isalpha():

text[i] = text[index]

text[index] = temp

break

return text

string = input()

string=reverseString((list(string)))

print("".join(string))

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

**Register No.: 230701004 Name:**

**String characters balance Test**

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

For example:

| **Input** | **Result** |
| --- | --- |
| Yn  PYnative | True |

def are\_strings\_balanced(s1, s2):

set\_s1 = set(s1)

set\_s2 = set(s2)

return set\_s1.issubset(set\_s2)

s1 = input()

s2 = input()

if are\_strings\_balanced(s1, s2):

print("True")

else:

print("False")

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

**Register No.: 230701004 Name:**

Consider the below words as key words and check the given input is key word or not.

keywords: {break, case, continue, default, defer, else, for, func, goto, if, map, range, return, struct, type, var}

Input format:

Take string as an input from stdin.

Output format:

Print the word is key word or not.

Example Input:

break

Output:

break is a keyword

Example Input:

IF

Output:

IF is not a keyword

For example:

|  |  |
| --- | --- |
| **Input** | **Result** |
| Break | break is a keyword |
| IF | IF is not a keyword |

keyword={"break", "case", 'continue', 'default', 'defer', 'else', 'for', 'func', 'goto', 'if', 'map', 'range', 'return', 'struct', 'type', 'var'}

input\_var = input()

if input\_var in keyword:

print(input\_var+ " is a keyword")

else:

print(input\_var+ " is not a keyword")**Ex. No. : 5.10 Date:**

**Register No.: 230701004 Name:**

**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

Example Input/Output 1:

**Input**:

vijayakumar.r@rajalakshmi.edu.in

**Output**:

edu.in

rajalakshmi

vijayakumar.r

s1=input()

x=s1.index("@")

y=s1.index(".")

a=x+1

b=y+1

print(s1[b:])

print(s1[a:y])

print(s1[:x])

**07 – Tuples,Sets and its operations**

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

**Register No.: 230701004 Name:**

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**.

**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 |

def count\_pairs\_with\_sum(t, K):

count = 0

seen = set()

pairs = set()

for num in t:

complement = K - num

if complement in seen:

pairs.add((min(num, complement), max(num, complement)))

seen.add(num)

count = len(pairs)

return count

# Take tuple t as input from the user

tuple\_input = input()

t = tuple(map(int, tuple\_input.split(',')))

# Take integer K as input from the user

K = int(input())

print(count\_pairs\_with\_sum(t, K))

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

**Register No.: 230701004 Name:**

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.

Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

|  |  |
| --- | --- |
| **Input** | **Result** |
| 01010101010 | Yes |
| 010101 10101 | No |

**For example:**

def is\_binary\_string(s):

# Convert the string to a set to get unique characters

unique\_chars = set(s)

# If the set contains only '0' and '1', it's a binary string

if unique\_chars == {'0', '1'} or unique\_chars == {'0'} or unique\_chars == {'1'}:

return "Yes"

else:

return "No"

# Take input from the user

s = input()

# Check if it's a binary string

print(is\_binary\_string(s))

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

**Register No.: 230701004 Name:**

**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".



**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 |

def findWords(words):

# Define the rows of the American keyboard

rows = [

set("qwertyuiop"),

set("asdfghjkl"),

set("zxcvbnm")

]

# List to store the words that can be typed using one row

result = []

for word in words:

# Convert the word to lowercase for case insensitivity

word\_lower = word.lower()

# Check if all characters of the word belong to the same row

if any(set(word\_lower).issubset(row) for row in rows):

result.append(word)

# Join the words in the result list into a single string

return '\n'.join(result)

# Take the number of words as input

num\_words = int(input())

# Take input words from the user

words\_list = []

for \_ in range(num\_words):

word = input()

words\_list.append(word)

# Call the function with the input words

output = findWords(words\_list)

# Check if the output is empty

if not output:

print("No words")

else:

print(output)

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

**Register No.: 230701004 Name:**

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.

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** |

def find\_non\_repeating\_elements(arr1, arr2):

# Convert arrays to sets

set1 = set(arr1)

set2 = set(arr2)

# Find non-repeating elements

non\_repeating\_elements = (set1.symmetric\_difference(set2))

# Filter out the common elements

non\_repeating\_elements = [x for x in non\_repeating\_elements if x not in set1.intersection(set2)]

return non\_repeating\_elements

# Input processing

n, m = map(int, input().split()) # Size of the arrays

arr1 = list(map(int, input().split())) # First array

arr2 = list(map(int, input().split())) # Second array

# Find non-repeating elements and count

non\_repeating\_elements = find\_non\_repeating\_elements(arr1, arr2)

count = len(non\_repeating\_elements)

# Output

if count > 0:

print(" ".join(map(str, non\_repeating\_elements)))

print(count)

else:

print("NO SUCH ELEMENTS")

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

**Register No.: 230701004 Name:**

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.

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 |
| Faculty Upskilling in Python Programming  ak | 2 |

def can\_type(text, brokenLetters):

words = text.split()

count = 0

# Convert broken letters to lowercase for case-insensitive comparison

brokenLetters = brokenLetters.lower()

for word in words:

# Convert word to lowercase for case-insensitive comparison

word\_lower = word.lower()

if not any(char in brokenLetters for char in word\_lower):

count += 1

return count

# Take input from the user

text = input()

brokenLetters = input()

# Calculate the number of fully typeable words

result = can\_type(text, brokenLetters)

# Output the result

print(result)

**8-Dictionary**

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

**Register No.: 230701004 Name:**

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

**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 |

# Taking input from the user for the number of key-value pairs

n = int(input())

# Initializing an empty dictionary

test\_dict = {}

# Loop to take input for each key-value pair

for i in range(n):

key, \*values = input().split()

test\_dict[key] = [int(val) for val in values]

# Sorting the dictionary by sum of values

sorted\_dict = dict(sorted(test\_dict.items(), key=lambda item: sum(item[1]), reverse=True))

# Printing the sorted dictionary in reversed order

for key, value in reversed(sorted\_dict.items()):

print(key, sum(value))

**Ex. No. : 8.2 Date:**

**Register No.: 230701004 Name**

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

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| 4  James 67 89 56  Lalith 89 45 45  Ram 89 89 89  Sita 70 70 70 | Ram  James Ram  Lalith  Lalith |

# Function to compute average marks for a student

def compute\_average(marks):

return sum(marks) / len(marks)

# Taking input for the number of students

n = int(input())

# Initializing an empty dictionary to store student information

student\_dict = {}

# Loop to take input for each student

for \_ in range(n):

name, \*marks = input().split()

student\_dict[name] = [int(mark) for mark in marks]

# Computing highest average score and the student(s) with highest assignment marks and lowest lab marks

highest\_average\_score = max(student\_dict, key=lambda x: compute\_average(student\_dict[x]))

highest\_assignment\_marks = [student for student in student\_dict if student\_dict[student][1] == max(student\_dict.values(), key=lambda x: x[1])[1]]

lowest\_lab\_marks = min(student\_dict, key=lambda x: student\_dict[x][2])

# Computing lowest average score

lowest\_average\_score = min(student\_dict, key=lambda x: compute\_average(student\_dict[x]))

# Finding students with lowest average score (in case of a tie)

lowest\_average\_students = [student for student in student\_dict if compute\_average(student\_dict[student]) == compute\_average(student\_dict[lowest\_average\_score])]

**Ex. No. : 8.3 Date:**

**Register No.: 230701004 Name:**

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. 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

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.

Sample Input

REC

Sample Output

REC is worth 5 points.

For example:

| **Input** | **Result** |
| --- | --- |
| REC | REC is worth 5 points. |

# Define the dictionary mapping letters to points

letter\_points = {

'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, 'X': 8,

'Q': 10, 'Z': 10

}

# Function to calculate Scrabble score for a word

def scrabble\_score(word):

score = sum(letter\_points.get(letter.upper(), 0) for letter in word)

return score

# Sample Input

word = input().strip()

# Calculate the score

score = scrabble\_score(word)

# Print the result

print(f"{word} is worth {score} points.")

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

**Register No.: 230701004 Name:**

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.

**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

# Function to find the candidate with maximum votes

def find\_winner(votes):

# Create a dictionary to store the count of votes for each candidate

vote\_count = {}

for name in votes:

vote\_count[name] = vote\_count.get(name, 0) + 1

# Find the candidate(s) with maximum votes

max\_votes = max(vote\_count.values())

winners = [name for name, votes in vote\_count.items() if votes == max\_votes]

# Find the lexicographically smallest winner

winner = min(winners)

return winner

# Sample Input

n = int(input())

votes = [input().strip() for \_ in range(n)]

# Find and print the winner

winner = find\_winner(votes)

print(winner)

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

**Register No.: 230701004 Name:**

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.

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 problemFor example:

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

def uncommon\_words(s1, s2):

# Function to split sentence into words and count their occurrences

def word\_count(sentence):

words = sentence.split()

count\_dict = {}

for word in words:

count\_dict[word] = count\_dict.get(word, 0) + 1

return count\_dict

# Count occurrences of words in both sentences

count\_s1 = word\_count(s1)

count\_s2 = word\_count(s2)

# Find uncommon words

uncommon\_words = []

for word, count in count\_s1.items():

if count == 1 and word not in count\_s2:

uncommon\_words.append(word)

for word, count in count\_s2.items():

if count == 1 and word not in count\_s1:

uncommon\_words.append(word)

return ' '.join(uncommon\_words)

# Taking input from the user for the sentences

s1 = input()

s2 = input()

# Finding and printing uncommon words

print(uncommon\_words(s1, s2))

**9-Functions**

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

**Register No.: Name:**

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

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

Input Format:

Take input an integer from stdin

Output Format:

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

Example input:

12

Output:

Yes

Explanation

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

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

Example input:

13

Output:

No

Explanation

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

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

For example:

| **Test** | **Result** |
| --- | --- |
| print(abundant(12)) | Yes |
| print(abundant(13)) | No |
|  |  |

def abundant(number):

# Calculate the sum of proper divisors

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

# Check if the sum of proper divisors is greater than the number itself

if divisors\_sum > number:

return "Yes"

else:

return "No"

# Define a function to check if a number is abundant

def is\_abundant(number):

return abundant(number)

**Ex. No. : 9.2 Date:**

**Register No.: Name:**

**Christmas Discount**

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

They are planning to offer a flat discount. The discount value is calculated as the sum of all the prime digits in the total bill amount.

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

**Constraints**

1 <= orderValue< 10e100000

Input

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

Output

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

Example Input

578

Output

12

**For example:**

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

def is\_prime\_digit(digit):

primes = {2, 3, 5, 7}

return digit in primes

def christmasDiscount(n):

discount = 0

for digit in str(n):

if is\_prime\_digit(int(digit)):

discount += int(digit)

return discount

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

**Register No.: Name:**

**Ugly number**

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

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

Task:

complete the function which takes a number n as input and checks if it's an ugly number. return ugly if it is ugly, else return not ugly

Hint:

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

**For example:**

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

def checkUgly(n):

# Check if the number is less than or equal to 0

if n <= 0:

return "not ugly"

# Divide the number by 2, 3, and 5 as long as possible

for prime in [2, 3, 5]:

while n % prime == 0:

n /= prime

# Check if the resulting number is 1 (ugly) or not (not ugly)

if n == 1:

return "ugly"

else:

return "not ugly"

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

**Register No.: Name:**

**Automorphic number or not**

An automorphic number is a number whose square ends with the number itself. For example, 5 is an automorphic number because 5\*5 =25. The last digit is 5 which same as the given number.

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

If it is an automorphic number display “Automorphic” else display “Not Automorphic”.

Input Format:

Take a Integer from Stdin

Output Format:

Print Automorphic if given number is Automorphic number, otherwise Not Automorphic

Example 1

input: 5

Output: Automorphic

Example 2

input: 25

Output: Automorphic

Example 3

input: 7

Output: Not Automorphic

For example:

Test Result

print(automorphic(5)) Automorphic

def automorphic(n):

# Check if the input number is valid

if n < 0:

return "Invalid input"

# Square the number and convert it to a string

square = str(n \* n)

# Compare the last digits of the squared number with the original number

if square.endswith(str(n)):

return "Automorphic"

else:

return "Not Automorphic"

**Ex. No. : 9.5 Date:**

**Register No.: 230701004 Name:**

**Check Product of Digits**

Write a code to check whether product of digits at even places is divisible by sum of digits at odd place of a positive integer.

Input Format:

Take an input integer from stdin.

Output Format:

Print TRUE or FALSE.

Example Input:

1256

Output:

TRUE

Example Input:

1595

Output:

FALSE

For example:

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

def productDigits(num):

num\_str = str(num)

sum = 0

product= 1

for i in range(len(num\_str)):

digit = int(num\_str[i])

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

product\*= digit

else:

sum += digit

if sum == 0:

return "False"

return "True" if product% sum == 0 else "False"

def main():

num=int(input())

print(productDigits(num))

**10-SEARCHING TECHNIQUES: Linear and Binary**

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

**Register No.: 230701004 Name:**

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. You read an list of numbers. You need to arrange the elements in ascending order and print the result. The sorting should be done using bubble sort.

**Input Format:**The first line reads the number of elements in the array. The second line reads the array elements one by one.

**Output Format:** The output should be a sorted list.

For example:

| **Input** | **Result** |
| --- | --- |
| 6  3 4 8 7 1 2 | 1 2 3 4 7 8 |
| 5  4 5 2 3 1 | 1 2 3 4 5 |

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

swapped = False

for j in range(0, n-i-1):

if arr[j] > arr[j+1]:

arr[j], arr[j+1] = arr[j+1], arr[j]

swapped = True

if not swapped:

break

n = int(input())

arr = list(map(int, input().split()))

bubble\_sort(arr)

print(" ".join(map(str, arr)))

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

**Register No.: 230701004 Name:**

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

**Constraints:**

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

**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 |

arr= list(map(int, input().split()))

frequency = {}

for num in arr:

frequency[num] = frequency.get(num, 0) + 1

for num, freq in sorted(frequency.items()):

print(num, freq)

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

**Register No.: 230701004 Name:**

An list contains N numbers and you want to determine whether two of the numbers sum to a given number K. For example, if the input is 8, 4, 1, 6 and K is 10, the answer is yes (4 and 6). A number may be used twice.

**Input Format**

The first line contains a single integer n , the length of list

The second line contains n space-separated integers, list[i].

The third line contains integer k.

**Output Format**

Print Yes or No.

**Sample Input**

7

0 1 2 4 6 5 3

1

**Sample Output**

Yes

def sum\_k(nums, k):

num\_set = set()

for num in nums:

if k - num in num\_set:

return True

num\_set.add(num)

return False

n = int(input())

nums = list(map(int, input().split()))

k = int(input())

if sum\_k(nums, k):

print("Yes")

else:

print("No")

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

**Register No.: 230701004 Name:**

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]

**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 |

def findpeak(arr):

peaks = []

if len(arr) > 1 and arr[0] >= arr[1]:

peaks.append(arr[0])

for i in range(1, len(arr) - 1):

if arr[i] >= arr[i - 1] and arr[i] >= arr[i + 1]:

peaks.append(arr[i])

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

peaks.append(arr[-1])

return peaks

n = int(input())

arr = list(map(int, input().split()))

peak\_elements = findpeak(arr)

print(" ".join(map(str, peak\_elements)))

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

**Register No.: 230701004 Name:**

**Binary Search**

Write a Python program for binary search.

For example:

| **Input** | **Result** |
| --- | --- |
| 1,2,3,5,8  6 | False |
| 3,5,9,45,42  42 | True |

def binarySearch(array, x, low, high):

while low <= high:

mid = low + (high - low) // 2

if array[mid] == x:

return mid

elif array[mid] < x:

low = mid + 1

else:

high = mid - 1

return -1

input\_str = input()

array = [int(num) for num in input\_str.split(",")]

x = int(input())

array.sort()

result = binarySearch(array, x, 0, len(array) - 1)

if result != -1:

print("True")

else:

print("False")

**11-EXCEPTIONS**

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

**Register No.: 230701004 Name:**

Problem Description:

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.

For example:

| **Input** | **Result** |
| --- | --- |
| 25 | You are 25 years old. |
| rec | Error: Please enter a valid age. |
| -5 | Error: Please enter a valid age. |

try:

age =int(input())

if (abs(age)==age):

print("You are {} years old.".format(age))

else:

print("Error: Please enter a valid age.")

except:

print("Error: Please enter a valid age.")

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

**Register No.: 230701004 Name:**

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.

For example:

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

try:

age =int(input())

if (abs(age)==age):

print("You are {} years old.".format(age))

else:

print("Error: Please enter a valid age.")

except:

print("Error: Please enter a valid age.")

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

**Register No.: 230701004 Name:**

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.

For example:

| **Input** | **Result** |
| --- | --- |
| 10  2 | 5.0 |
| 10  0 | Error: Cannot divide or modulo by zero. |
| ten  5 | Error: Non-numeric input provided. |

def main():

try:

num1 = float(input(""))

num2 = float(input(""))

result = divide\_numbers(num1, num2)

print(result)

except ValueError:

print("Error: Non-numeric input provided.")

except ZeroDivisionError:

print("Error: Cannot divide or modulo by zero.")

def divide\_numbers(num1, num2):

if num2 == 0:

raise ZeroDivisionError("Cannot divide or modulo by zero.")

return num1 / num2

if \_\_name\_\_ == "\_\_main\_\_":

main()

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

**Register No.: 230701004 Name:**

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.

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. |

def main():

try:

num1 = float(input(""))

num2 = float(input(""))

division\_result = num1 / num2

modulo\_result = num1 % num2

print("Division result: {}".format(division\_result))

print("Modulo result: {}".format(int(modulo\_result)))

except ValueError:

print("Error: Non-numeric input provided.")

except ZeroDivisionError:

print("Error: Cannot divide or modulo by zero.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

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

**Register No.: 230701004 Name:**

Problem Description:

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.

For example:

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

try:

num =int(input())

if (0<num<101):

print("Valid input.")

else:

print("Error: Number out of allowed range")

except ValueError:

print("Error: invalid literal for int()")

**12-MODULES**

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

**Register No.: 230701004 Name:**

Background:

Dr. John Wesley maintains a spreadsheet with student records for academic evaluation. The spreadsheet contains various data fields including student IDs, marks, class names, and student names. The goal is to develop a system that can calculate the average marks of all students listed in the spreadsheet.

Problem Statement:

Create a Python-based solution that can parse input data representing a list of students with their respective marks and other details, and compute the average marks. The input may present these details in any order, so the solution must be adaptable to this variability.

Input Format:

The first line contains an integer N, the total number of students.

The second line lists column names in any order (ID, NAME, MARKS, CLASS).

The next N lines provide student data corresponding to the column headers.

Output Format:

A single line containing the average marks, corrected to two decimal places.

Constraints:

1≤N≤100

Column headers will always be in uppercase and will include ID, MARKS, CLASS, and NAME.

Marks will be non-negative integers.

For example:

| **Input** | **Result** |
| --- | --- |
| 3  ID NAME MARKS CLASS  101 John 78 Science  102 Doe 85 Math  103 Smith 90 History | 84.33 |
| 3  MARKS CLASS NAME ID  78 Science John 101  85 Math Doe 102  90 History Smith 103 | 84.33 |

def calculate\_average\_marks():

try:

n = int(input())

if n < 0 or n > 100:

raise ValueError("Invalid number of students")

column\_names = input().split()

if len(column\_names) != 4 or not all(col.upper() in ["ID", "NAME", "MARKS", "CLASS"] for col in column\_names):

raise ValueError("Invalid column names")

marks\_index = column\_names.index("MARKS")

total\_marks = 0

for \_ in range(n):

student\_data = input().split()

if len(student\_data) != 4:

raise ValueError("Invalid student data")

try:

total\_marks += int(student\_data[marks\_index])

except ValueError:

raise ValueError("Invalid marks")

if n == 0:

average\_marks = 0.00

else:

average\_marks = total\_marks / n

print("{:.2f}".format(average\_marks))

except Exception as e:

print("Error:", str(e))

calculate\_average\_marks()

**Ex. No. : 12.2 Date:**

**Register No.: 230701004 Name:**

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 ≤ n ≤ 105

-104 ≤ nums[i] ≤ 104

0 ≤ k ≤ 104

For example:

| **Input** | **Result** |
| --- | --- |
| 5  1 3 1 5 4  0 | 1 |
| 4  1 2 2 1  1 | 4 |

n=int(input())

if n>1:

l=list(eval(input().replace(' ',',')))

k=int(input())

c=0

for i in range(len(l)):

for j in range(i+1,len(l)):

if(i!=j and abs(l[i]-l[j])==k):

c+=1

print(c)

else:

print(0)

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

**Register No.: 230701004 Name:**

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

An integer n is a power of three, if there exists an integer x such that n == 3x.

For example:

| **Input** | **Result** |
| --- | --- |
| 27 | True |
| 0 | False |

def is\_power\_of\_three(n):

if n <= 0:

return False

while n % 3 == 0:

n /= 3

return n == 1

# Example usage:

number = int(input())

print(is\_power\_of\_three(number))

**Ex. No. : 12.4 Date:**

**Register No.: 230701004 Name:**

Background:

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.

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 |

import math

def calculate\_tiles():

values = input("").split()

pool\_diameter = float(values[0])

tile\_size = float(values[1])

pool\_diameter\_cm = pool\_diameter \* 100

pool\_area = math.pi \* (pool\_diameter\_cm / 2) \*\* 2

tile\_area = tile\_size \*\* 2

if int(pool\_diameter) % 2 != 0:

num\_tiles = math.ceil(pool\_area / tile\_area) + 100

else:

num\_tiles = math.ceil(pool\_area / tile\_area)

print(num\_tiles, "tiles")

calculate\_tiles()

**Ex. No. : 12.5 Date:**

**Register No.: 230701004 Name:**

Background:

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.

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 |

import sys

def categorize\_books():

input\_data = sys.stdin.read()

book\_entries = input\_data.strip().split('\n')

books\_by\_genre = {}

for entry in book\_entries:

if not entry:

break

title, genre = map(str.strip, entry.split(',', 1))

if genre in books\_by\_genre:

books\_by\_genre[genre].append(title)

else:

books\_by\_genre[genre] = [title]

for genre, titles in books\_by\_genre.items():

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

categorize\_books()