**RAJALAKSHMI ENGINEERING COLLEGE**

**RAJALAKSHMI NAGAR, THANDALAM – 602 105**



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

# Name: AKSHAY VENKAT KRISHNA

# Year/Branch/Section:2023-24/Computer Science Engineering/A

# Register No.: 230701022

# Semester: II

# Academic Year: 2023-24

INDEX

**Register No.: 230701022 Name: Akshay Venkat Krishna**

**Year: 2023-24 Branch: CSE**

**Section: ‘A’**

| **S.**  **No.** | **Date** | **Title** | **Page**  **No.** | **Teacher’s Signature /**  **Remarks** |
| --- | --- | --- | --- | --- |
| **Introduction to python-Variables-Datatypes-Input/Output-Formatting** | | | | |
| 1.1 | **19/03/24** | Convert strings | **7** |  |
| 1.2 | **19/03/24** | Basic salary | **8** |  |
| 1.3 | **19/03/24** | Square Root | **9** |  |
| 1.4 | **19/03/24** | Gain percent | **10** |  |
| 1.5 | **19/03/24** | Deposits | **11** |  |
| 1.6 | **19/03/24** | Carpenter | **12** |  |
| **Operators in Python** | | | | |
| 2.1 | **19/03/24** | Meal Order | **14** |  |
| 2.2 | **19/03/24** | Savings | **15** |  |
| 2.3 | **19/03/24** | Binary | **16** |  |
| 2.4 | **19/03/24** | Rotract Club | **17** |  |
| 2.5 | **19/03/24** | Gizmo | **18** |  |
| 2.6 | **19/03/24** | Doll show | **19** |  |
| 2.7 | **19/03/24** | Hercules | **20** |  |
| 2.8 | **19/03/24** | Chocolates | **21** |  |
| 2.9 | **19/03/24** | Last digit of given number | **24** |  |
| 2.10 | **19/03/24** | C and D | **25** |  |
| **Selection Structures in Python** | | | | |
| 3.1 | **26/03/24** | Chinese Zodiac | **27** |  |
| 3.2 | **26/03/24** | Second Last Digit | **29** |  |
| 3.3 | **26/03/24** | Electricity Bill | **30** |  |
| 3.4 | **26/03/24** | Month name to Days | **31** |  |
| 3.5 | **26/03/24** | IN/OUT | **32** |  |
| 3.6 | **26/03/24** | Admission Criteria | **33** |  |
| 3.7 | **26/03/24** | Leap Year | **34** |  |
| 3.8 | **26/03/24** | Classification of Triangles | **35** |  |
| 3.9 | **26/03/24** | Pythagorean Triplets | **36** |  |
| 3.10 | **26/03/24** | Vowel/Consonant | **45** |  |
| **Algorithmic Approach: Iteration Control Structures** | | | | |
| 4.1 | **26/03/24** | Number of Primes | **47** |  |
| 4.2 | **26/03/24** | Non-Repeated Digits Count | **49** |  |
| 4.3 | **26/03/24** | Prime Checking | **51** |  |
| 4.4 | **26/03/24** | Next Perfect Square | **55** |  |
| 4.5 | **26/03/24** | Nth Fibonacci | **56** |  |
| 4.6 | **26/03/24** | Disarium Number | **58** |  |
| 4.7 | **26/03/24** | Sum of Series | **60** |  |
| 4.8 | **26/03/24** | Unique Digits Count | **61** |  |
| 4.9 | **26/03/24** | Product of single digits | **63** |  |
| 4.10 | **26/03/24** | Perfect Square After adding One | **68** |  |
| **Strings in Python** | | | | |
| 5.1 | **16/04/24** | Count chars | **70** |  |
| 5.2 | **16/04/24** | Decompress the String | **73** |  |
| 5.3 | **16/04/24** | First N Common Characters | **75** |  |
| 5.4 | **16/04/24** | Remove Characters | **78** |  |
| 5.5 | **16/04/24** | Remove Palindrome Words | **79** |  |
| 5.6 | **16/04/24** | Return Second Word in Uppercase | **80** |  |
| 5.7 | **16/04/24** | Reverse String | **81** |  |
| 5.8 | **16/04/24** | String characters balance Test | **82** |  |
| 5.9 | **16/04/24** | Unique Names | **83** |  |
| 5.10 | **16/04/24** | Username Domain Extension | **84** |  |
| **List in Python** | | | | |
| 6.1 | **02/05/24** | Monotonic array | **86** |  |
| 6.2 | **02/05/24** | Check pair with difference k . | **88** |  |
| 6.3 | **02/05/24** | Count Elements | **90** |  |
| 6.4 | **02/05/24** | Distinct Elements in an Array | **92** |  |
| 6.5 | **02/05/24** | Element Insertion | **94** |  |
| 6.6 | **02/05/24** | Find the Factor | **96** |  |
| 6.7 | **02/05/24** | Merge list | **98** |  |
| 6.8 | **02/05/24** | Merge Two Sorted Arrays Without Duplication | **103** |  |
| 6.9 | **02/05/24** | Print Element Location | **106** |  |
| 6.10 | **02/05/24** | Strictly increasing | **108** |  |
| **Tuples & Set** | | | | |
| 7.1 | **27/05/24** | Binary String | **110** |  |
| 7.2 | **27/05/24** | Check Pair | **111** |  |
| 7.3 | **27/05/24** | DNA Sequence | **112** |  |
| 7.4 | **27/05/24** | Print repeated no | **114** |  |
| 7.5 | **27/05/24** | Remove repeated | **115** |  |
| 7.6 | **27/05/24** | malfunctioning keyboard | **117** |  |
| 7.7 | **27/05/24** | American keyboard | **119** |  |
| **Dictionary** | | | | |
| 8.1 | **28/05/24** | Uncommon Words | **121** |  |
| 8.2 | **28/05/24** | Sort Dictionary By Values Summation | **124** |  |
| 8.3 | **28/05/24** | Winner Of Election | **127** |  |
| 8.4 | **28/05/24** | Student Record | **129** |  |
| 8.5 | **28/05/24** | Scramble Score | **133** |  |
| **Functions** | | | | |
| 9.1 | **31/05/24** | Abundant Number | **135** |  |
| 9.2 | **31/05/24** | Automorphic number or not | **136** |  |
| 9.3 | **31/05/24** | Check Product of Digits | **137** |  |
| 9.4 | **31/05/24** | Christmas Discount | **138** |  |
| 9.5 | **31/05/24** | Coin Change | **139** |  |
| 9.6 | **31/05/24** | Difference Sum | **140** |  |
| 9.7 | **31/05/24** | Ugly number | **142** |  |
| **Searching & Sorting** | | | | |
| 10.1 | **05/06/24** | Merge Sort | **144** |  |
| 10.2 | **05/06/24** | Bubble Sort | **146** |  |
| 10.3 | **05/06/24** | Peak Element | **148** |  |
| 10.4 | **05/06/24** | Binary Search | **150** |  |
| 10.5 | **05/06/24** | Frequency of Numbers | **152** |  |

| **Exceptions** | | | | |
| --- | --- | --- | --- | --- |
| 11.1 | **09/06/24** | Merge Sort | **154** |  |
| 11.2 | **09/06/24** | Bubble Sort | **156** |  |
| 11.3 | **09/06/24** | Peak Element | **158** |  |
| 11.4 | **09/06/24** | Binary Search | **160** |  |
| 11.5 | **09/06/24** | Frequency of Numbers | **164** |  |

| **Modules** | | | | |
| --- | --- | --- | --- | --- |
| 21.1 | **09/06/24** | Merge Sort | **166** |  |
| 12.2 | **09/06/24** | Bubble Sort | **168** |  |
| 12.3 | **09/06/24** | Peak Element | **171** |  |
| 12.4 | **09/06/24** | Binary Search | **172** |  |
| 12.5 | **09/06/24** | Frequency of Numbers | **173** |  |

**1- Introduction to Python-Variables-Datatypes**

**Input/Output-Formatting**

**Ex. No. : 1.1 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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'> |

**d=int(input());**

**c=float(input());**

**print(d,",",type(d),sep="")**

**print(round(c,1),",",type(c),sep="")**

**Ex. No. : 1.2 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 |

*sal=int(input())*

*print(int((sal+(.6\*sal))))*

**Ex. No. : 1.3 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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**

**num=float(input())**

**num=math.sqrt(num)**

**print("{:.3f}".format(num))**

**Ex. No. : 1.4 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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())**

**g=z-(x+y)**

**gp=(g/(x+y))\*100**

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

**Ex. No. : 1.5 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. |

l=float(input())

L=float(input())

num=(l\*.1)+(L\*.25)

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

**Ex. No. : 1.6 Date: 19/03/24**

**Register No.: 230701022 Name:Akshay Venkat Krishna**



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

**For example:**

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

sal=int(input())

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

num=10+n

print("weekdays","{:.2f}".format(num),"\nweekend","{:.2f}".format(n))

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

**Ex. No. : 2.1 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**For example:**

| **Input** | **Result** |
| --- | --- |
| 100 | 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. : 2.2 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 10000 | 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. : 2.3 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 2.4 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Note:

Dont use if-else. Operators alone must be used .

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| 18  40 | False |

x=int(input())

y=int(input())

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

print("True")

else:

print("False")

**Ex. No. : 2.5 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Input:

10

20

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Output:

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. : 2.6 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**For example:**

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

**x=int(input())**

**if (x>0 and x<=100):**

**if(x%2==0):**

**print("True")**

**else:**

**print("False")**

**else:**

**print("False")**

**Ex. No. : 2.7 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**For example:**

| **Input** | **Result** |
| --- | --- |
| 32  43 | False |

**x=int(input())**

**y=int(input())**

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

**print("True")**

**else:**

**print("False")**

**Ex. No. : 2.8 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 2.9 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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** |
| --- | --- |
| 197 | 7 |
| -197 | 7 |

n=int(input())

n=abs(n)

print(n%10)

**Ex. No. : 2.10 Date: 19/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna.**



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 or x==1):

if (x==0):

print("C")

else:

print("D")

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

**Ex. No. : 3.1 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.")**

**Ex. No. : 3.2 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 |
| 5 | -1 |

**num=int(input())**

**num=abs(num)**

**n=num//10**

**if(num<10 and num>0):**

**print(-1)**

**else:**

**print(n%10)**

**Ex. No. : 3.3 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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** |
| --- | --- |
| 100.00 | 120.00 |

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: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. |

**month=input()**

**if (month=="February"):**

**print("February has 28 or 29 days in it.")**

**elif(month=="April"):**

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

**else:**

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

**Ex. No. : 3.5 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



IN / OUT

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

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 |

num=int(input())

n=int(input())

if(n>(num//2)):

print("IN")

else:

print("OUT")

**Ex. No. : 3.6 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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** |
| --- | --- |
| 70  60  80 | The candidate is eligible |

p=int(input())

c=int(input())

m=int(input())

sum=p+c+m

if p>=55 and c>=50 and m>=65:

print("The candidate is eligible")

elif sum>=180:

print("The candidate is eligible")

else:

print("The candidate is not eligible")

**Ex. No. : 3.7 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

• Any year that is divisible by 400 is a leap year.

• Of the remaining years, any year that is divisible by 100 is not a leap year.

• Of the remaining years, any year that is divisible by 4 is a leap year.

• All other years are not leap years.

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

Sample Input 1

1900

Sample Output 1

1900 is not a leap year.

Sample Input 2

2000

year = int(input())

if (year % 400 == 0) and (year % 100 == 0):

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

elif (year % 4 ==0) and (year % 100 != 0):

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

else:

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

**Ex. No. : 3.8 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

Sample Input 2

40

40

80

Sample Output 2

That's a isosceles triangle

Sample Input 3

50

60

70

Sample Output 3

That's a scalene triangle

**s1=int(input())**

**s2=int(input())**

**s3=int(input())**

**if(s1==s2 and s1==s3):**

**print("That's a equilateral triangle")**

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

**print("That's a isosceles triangle")**

**else:**

**print("That's a scalene triangle")**

**Ex. No. : 3.9 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

For example, 3, 5 and 4 form a Pythagorean triple, since 3\*3 + 4\*4 = 25 = 5\*5

You are given three integers, a, b, and c. They need not be given in increasing order. If they form a Pythagorean triple, then print "yes", otherwise, print "no". Please note that the output message is in small letters.

Sample Input

3

5

4

Sample Output

yes

Sample Test Cases

Test Case 1

Input

3

5

4

Output

yes

Test Case 2

Input

5

8

2

Output

no

n1=int(input())

n2=int(input())

n3=int(input())

if((n1\*n1+ n2\*n2)==n3\*n3):

print("yes")

elif((n2\*n2+ n3\*n3)==n1\*n1):

print("yes")

elif((n1\*n1+ n3\*n3)==n2\*n2):

print("yes")

else:

print("no")

**Ex. No. : 3.10 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.

**For example:**

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

a=input()

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

print("It's a vowel.")

elif(a=='y'):

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

else:

print("It's a consonant.")

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

**Ex. No. : 4.1 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**Write a program to find the count of the number of prime numbers in a specified range.**

**The starting and ending number of the range will be provided as input to the program.**

**Assumption: 2<starting number of the range ending number of the range-7919**

**Example1: if the starting and ending number or the range is given as 2 and 20, the program must return &, because there are 8 prime numbers in the specified range from 2 to 20 namely 2.3.5, 7, 11, 13, 17, 19)**

**Example: If the starting and ending number of the range is given as 700 and 725, the program must retum 3, because there are 3 prime numbers in the specified range from 700 to 725, namely (701, 709,719)**

**n-int(input())**

**N=int(input())**

**C=0**

**for num in range(n, N+1):**

**if num>1:**

**for i in range(2, num):**

**if(num%i)==0:**

**break**

**else:**

**C+-1**

**print(c)**

**Ex. No. : 4.2 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



In mathematics, the factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n. For example,

5!5 x 4 x 3 x 2 x 1 = 120

4!= 4 x 3 x 2 x 1 = 24

9!= 9 x 8 x 7x6x5x4x3x2x1=362880

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

The given number will be passed to the program as an input of type int.

The program is expected to calculate the factorial of the given number and return it as an int type.

Assumptions for this program:

The given input number will always be greater than or equal to 1.

n=int(input())

f = 1

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

f=f\*i

print(f)

**Ex. No. : 4.3 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.

**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. : 4.4 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**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 = 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. : 4.5 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**Rakesh loves playing with numbers. He took the Fibonacci senes and wants to find the sum of squares of the series until a given value. Write a code that implements his task**

**Output Format:**

**Display the sum of squares of the Fibonacci series until the Nth term.**

**Example Input:**

**9**

**Output:**

**1870**

**Explanation:**

**The numbers are: 11.235.8 1321 34**

**Sum of their squares is 1+1+4+9+25+64+169+441-1156=1870**

**n=int(input())**

**fibo [0]\* (n + 1)**

**fibo[0] = 0**

**fibo[1] =1**

**for i in range(2, n + 1):**

**sum ((fibo[0] fibo[0]) +(fibo[1] \* fibo[1]))**

**fibo[1] (fibo[i-1] +fibo[i-2])**

**sum += (fibo[i] \* fibo[i]);**

**print(sum)**

**Ex. No. : 4.6 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**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:**

**Print 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**

**n=int(input())**

**sum=0**

**- for i in range(1, n//2+1):**

**if(n%i==0):**

**sum+=i**

**else:**

**pass**

**- if(sum>n):**

**print("Yes")**

**- else:**

**print("No")**

**Ex. No. : 4.7 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**Write a program to find the count of the number of prime numbers in a specified range.**

**The starting and ending number of the range will be provided as input to the program.**

**Assumption: 2 <=starting number of the range<= ending number of the range<=7919**

**Example1: If the starting and ending number or the range is given as 2 and 20, the program must return 8, because there are 8 prime numbers in the specified range from 2 to 20. namely (2. 3. 5, 7, 11, 13, 17, 19)**

**Example2: If the starting and ending number of the range is given as 700 and 725, the program must return 3, because there are 3 prime numbers in the specified range from 700 to 725, namely (701, 709, 719)**

**n=int(input())**

**N=int(input())**

**c=0**

**for num in range(n, N+1):**

**if num>1:**

**for i in range(2,num):**

**if(num%i)==0:**

**break**

**else:**

**c+=1**

**print(c)**

**Ex. No. : 4.8 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**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**

**num=int(input())**

**flag = False**

**if num == 1:**

**print(1)**

**elif num > 1:**

**for i in range(2, num): if (num % і) == 0: flag = True**

**break**

**if flag:**

**print(1)**

**else:**

**print(2)**

**Ex. No. : 4.9 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**Write a program to return the nth number in the fibonacci series.**

**The value of N will be passed to the program 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.**

**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. : 4.10 Date: 26/03/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**Input Format:**

**Single integer input.**

**Output Format:**

**Yes or No.**

**Example Input:**

**24**

**Output:**

**Yes**

**Example Input:**

**26**

**Output:**

**No**

**n=int(input())**

**n+=1**

**if(int(n\*0.5)\*2 == n):**

**print("Yes")**

**else:**

**print("No")**

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

**Ex. No. : 5.1 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 |

def count\_characters(string):

# Initialize counts

letter\_count = 0

digit\_count = 0

special\_count = 0

# Iterate through each character in the string

for char in string:

# Check if the character is a letter

if char.isalpha():

letter\_count += 1

# Check if the character is a digit

elif char.isdigit():

digit\_count += 1

# If it's not a letter or digit, consider it as a special symbol

else:

special\_count += 1

return letter\_count, digit\_count, special\_count

# Take input from the user

input\_string = input()

# Count the characters

letter\_count, digit\_count, special\_count = count\_characters(input\_string)

# Print the counts

print(letter\_count)

print(digit\_count)

print(special\_count)

**Ex. No. : 5.2 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Find if a String2 is substring of String1. If it is, return the index of the first occurrence. else return -1.

**Sample Input 1**

thistest123string

123

**Sample Output 1**

8

**def find\_substring(string1, string2):**

**# Find the index of the substring in the main string**

**index = string1.find(string2)**

**# Return the index if found, otherwise return -1**

**return index if index != -1 else -1**

**# Take input from the user**

**string1 = input()**

**string2 = input()**

**# Find if string2 is a substring of string1 and get the index**

**index = find\_substring(string1, string2)**

**# Print the result**

**print(index)**

**Ex. No. : 5.3 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Robert  is having 2 strings consist of uppercase & lowercase english letters. Now he want to compare those two strings lexicographically. The letters' case does not matter, that is an uppercase letter is considered equivalent to the corresponding lowercase letter.

### **Input**

The first line contains **T**. Then **T** test cases follow.

Each test case contains a two lines contains a string. The strings' lengths range from 1 to 100 inclusive. It is guaranteed that the strings are of the same length and also consist of uppercase and lowercase Latin letters.

### **Output**

If the first string is less than the second one, print "-1".  
If the second string is less than the first one, print "1".  
If the strings are equal, print "0".  
Note that the letters' case is not taken into consideration when the strings are compared.

def compare\_strings(s1, s2):

s1 = s1.lower() # Convert s1 to lowercase

s2 = s2.lower() # Convert s2 to lowercase

if s1 < s2:

return -1

elif s1 > s2:

return 1

else:

return 0

# Take input from the user

T = int(input())

for \_ in range(T):

string1 = input().strip()

string2 = input().strip()

# Compare the strings and print the result

print(compare\_strings(string1, string2))

**Ex. No. : 5.4 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

first

second

first

third

second

then your program should display:

first

second

third

def display\_unique\_words():

unique\_words = []

entered\_words = set() # To check for duplicates

while True:

word = input().strip()

if word == "":

break

elif word not in entered\_words:

unique\_words.append(word)

entered\_words.add(word)

for word in unique\_words:

print(word)

# Call the function to execute the program

display\_unique\_words()

**Ex. No. : 5.5 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Given a string S, which contains several words, print the count C of the words whose length is atleast L. (You can include punctuation marks like comma, full stop also as part of the word length. Space alone must be ignored)

**Input Format:**

The first line contains S.  
The second line contains L.

**Output Format:**

The first line contains C

**Boundary Conditions:**

2 <= Length of S <= 1000

**Example Input/Output 1:**

Input:

During and after Kenyattas inauguration police elsewhere in the capital, Nairobi, tried to stop the opposition from holding peaceful demonstrations.  
5

Output:

13

Explanation:

The words of minimum length 5 are  
During  
after  
Kenyattas  
inauguration  
police  
elsewhere  
capital,  
Nairobi,  
tried  
opposition  
holding  
peaceful  
demonstrations.

**s1=input()**

**l=int(input())**

**c=0**

**s2=(s1.split())**

**for i in s2:**

**if len(i)>=l:**

**c+=1**

**print(c)**

**Ex. No. : 5.6 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

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

# Take input from the user

inp= input()

temp=inp.strip().lower()

# Check if the input is a keyword or not

if inp in keywords:

print(inp, "is a keyword")

else:

print(inp, "is not a keyword")Students can manually mark this item complete: 4 Monotonic Array

**Ex. No. : 5.7 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



String should contain only the words are not palindrome.

**Sample Input 1**

Malayalam is my mother tongue

**Sample Output 1**

is my mother tongue

**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. : 5.8 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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:

abcd@gmail.com

Output:

com  
gmail  
abcd

s1=input()

x=s1.index("@")

y=s1.index(".")

a=x+1

b=y+1

print(s1[b:])

print(s1[a:y])

print(s1[:x])

**Ex. No. : 5.9 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 5.10 Date: 16/04/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**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))

### **06 - Strings in Python**

**Ex. No. : 6.1 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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)**

E**x. No. : 6.2 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.

**Example**

n = 20

p = 3

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if p = 3, then 4 is returned. If p > 6, 0 would be returned.

**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 is returned as the answer.

**Sample Case 2**

**Sample Input 2**

1

1

**Sample Output 2**

1

**Explanation 2**

Factoring n = 1 results in {1}. The p = 1st factor of 1 is returned as the answer.

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. : 6.3 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.

**def find\_locations\_and\_count(arr, target):**

**locations = []**

**count = 0**

**for i, num in enumerate(arr, start=1):**

**if num == target:**

**locations.append(i)**

**count += 1**

**return locations, count**

**# Main function**

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

**n = int(input().strip()) # Number of elements in the list**

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

**target = int(input().strip()) # Element to search**

**locations, count = find\_locations\_and\_count(arr, target)**

**# Print locations**

**for loc in locations:**

**print(f"{target} is present at location {loc}.")**

**# Print count**

**if count==0:**

**print(f"{target} is not present in the array.")**

**else:**

**print(f"{target} is present {count} times in the array.")**

**Ex. No. : 6.4 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

# 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. : 6.5 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 6.6 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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()) # Number of elements in the array**

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

**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")**

Input:

A&B

Output:

B&A

Explanation: As we ignore '&' and

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

For example:

Input Result

A&x#

x&A#

**Ex. No. : 6.7 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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")**

**Ex. No. : 6.8 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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.

Example

Input

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

**For example:**

| **Input** | **Result** |
| --- | --- |
| 1  3  1  3  5  4 | 1 |
| 1  3  1  3  5  99 | 0 |

**Ex. No. : 6.9 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**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()**

Example Input/Output 1:

**Input**:

vijayakumar.r@rajalakshmi.edu.in

**Output**:

edu.in

rajalakshmi

vijayakumar.r

**Ex. No. : 6.10 Date: 02/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Find the intersection of two sorted arrays.

OR in other words,

Given 2 sorted arrays, find all the elements which occur in both the arrays.

Input Format

The first line contains T, the number of test cases. Following T lines contain:

1.      Line 1 contains N1, followed by N1 integers of the first array

2.      Line 2 contains N2, followed by N2 integers of the second array

Output Format

The intersection of the arrays in a single line

Example

Input:

1

3 10 17 57

6 2 7 10 15 57 246

Output:

10 57

t = int(input())

for \_ in range(t):

f = 0

n = int(input())

n1 = []

for \_ in range(n):

x = int(input())

n1.append(x)

m = int(input())

n2 = []

for \_ in range(m):

x = int(input())

n2.append(x)

common\_elements = sorted(list(set(n1) & set(n2)))

if common\_elements:

print(\*common\_elements, sep=' ')

else:

break;

**07 - Functions**

**Ex. No. : 7.1 Date: 27/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Input:

5 4

1 2 8 6 5

2 6 8 10

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Output:

1 5 10

3

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127)  Input:

5 5

1 2 3 4 5

1 2 3 4 5

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Output:

NO SUCH ELEMENTS

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

# Convert arrays to sets to find common elements

set1 = set(arr1)

set2 = set(arr2)

# Find common elements

common\_elements = set1 & set2

# Find non-repeating elements in both arrays

non\_repeating\_arr1 = [x for x in arr1 if x not in common\_elements]

non\_repeating\_arr2 = [x for x in arr2 if x not in common\_elements]

# Combine the non-repeating elements from both arrays

non\_repeating\_elements = non\_repeating\_arr1 + non\_repeating\_arr2

# Print results

if non\_repeating\_elements:

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

print(len(non\_repeating\_elements))

else:

print("NO SUCH ELEMENTS")

# Read input

sizes = input().strip().split()

size1 = int(sizes[0])

**Ex. No. : 7.2 Date: 27/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**Example 1:**

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

**Output:** 2

**Example 2:**

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

**Output:** 3

**For example:**

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

**Ex. No. : 7.3 Date: 27/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 |
| 2  adsfd  afd | adsfd  afd |

**def find\_words(words):**

**# Define the sets for each row of the keyboard**

**row1 = set("qwertyuiop")**

**row2 = set("asdfghjkl")**

**row3 = set("zxcvbnm")**

**def can\_be\_typed(word):**

**# Convert word to lowercase to handle case insensitivity**

**lower\_word = set(word.lower())**

**# Check if the word can be typed using letters from one row**

**return lower\_word <= row1 or lower\_word <= row2 or lower\_word <= row3**

**# Filter and return the words that can be typed using one row**

**return [word for word in words if can\_be\_typed(word)]**

**# Function to take a list of words as input from the user**

**def input\_words():**

**# Read input from the user**

**input\_str = input()**

**# Split the input string into a list of words**

**words = input\_str.split()**

**return words**

**# Example usage:**

**words = input\_words()**

**result = find\_words(words)**

**print(result)**

**Ex. No. : 7.4 Date: 27/05/24**

**Register No.: 230701022 Name:Akshay Venkat Krishna**



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

**For example:**

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

**def is\_binary\_string(s):**

**# Define the set of binary characters**

**binary\_chars = {'0', '1'}**

**# Convert the string to a set of characters**

**char\_set = set(s)**

**# Check if char\_set is a subset of binary\_chars**

**if char\_set <= binary\_chars:**

**return "Yes"**

**else:**

**return "No"**

**# Example usage:**

**str1 = input()**

**print(is\_binary\_string(str1)) # Output: Yes**

**Ex. No. : 7.5 Date: 27/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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\_distinct\_pairs\_with\_sum(t, K):

seen = set()

pairs = set()

for number in t:

complement = K - number

if complement in seen:

# Add the pair in a sorted order to ensure distinctness

pairs.add(tuple(sorted((number, complement))))

seen.add(number)

# Return the number of distinct pairs

return len(pairs)

input\_str=input()

elements = input\_str.split(',')

elements = [int(element.strip()) for element in elements]

t = tuple(elements)

K = int(input())

print(count\_distinct\_pairs\_with\_sum(t, K)) # Output: 2

**08 – Tuple/Set**

**Ex. No. : 8.1 Date: 28/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 problem

**For example:**

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

s1 = input()

s2 = input()

words1 = s1.split()

words2 = s2.split()

count1 = {}

count2 = {}

for word in words1:

if word in count1:

count1[word] += 1

else:

count1[word] = 1

for word in words2:

if word in count2:

count2[word] += 1

else:

count2[word] = 1

cc = {}

for word in count1:

cc[word] = cc.get(word, 0) + count1[word]

for word in count2:

cc[word] = cc.get(word, 0) + count2[word]

result = ''

for word in cc:

if cc[word] == 1:

if (word in count1 and count1[word] == 1) or (word in count2 and count2[word] == 1):

result+=word +" "

print(result)

**Ex. No. : 8.2 Date: 28/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

def find\_winner(votes):

vote\_count = {}

for vote in votes:

if vote in vote\_count:

vote\_count[vote] += 1

else:

vote\_count[vote] = 1

max\_votes = 0

winner = ""

for candidate in vote\_count:

if vote\_count[candidate] > max\_votes or (vote\_count[candidate] == max\_votes and candidate < winner):

max\_votes = vote\_count[candidate]

winner = candidate

return winner

n=int(input())

votes=[]

for i in range(n):

votes.append(input())

print(find\_winner(votes)) # Output: john

**Ex. No. : 8.3 Date: 28/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

def get\_student\_data():

n = int(input())

student\_data = {}

for \_ in range(n):

data = input().split()

name = data[0]

marks = list(map(int, data[1:]))

student\_data[name] = marks

return student\_data

def identify\_highest\_average(student\_data):

highest\_avg = -1

highest\_avg\_students = []

for student, marks in student\_data.items():

avg = sum(marks) / len(marks)

if avg > highest\_avg:

highest\_avg = avg

highest\_avg\_students = [student]

elif avg == highest\_avg:

highest\_avg\_students.append(student)

return highest\_avg\_students

def identify\_highest\_assignment(student\_data):

highest\_assignment = -1

highest\_assignment\_students = []

for student, marks in student\_data.items():

if marks[1] > highest\_assignment:

highest\_assignment = marks[1]

highest\_assignment\_students = [student]

elif marks[1] == highest\_assignment:

highest\_assignment\_students.append(student)

return highest\_assignment\_students

def identify\_lowest\_lab(student\_data):

lowest\_lab = float('inf')

lowest\_lab\_students = []

for student, marks in student\_data.items():

if marks[2] < lowest\_lab:

lowest\_lab = marks[2]

lowest\_lab\_students = [student]

elif marks[2] == lowest\_lab:

lowest\_lab\_students.append(student)

return sorted(lowest\_lab\_students)

def identify\_lowest\_average(student\_data):

lowest\_avg = float('inf')

lowest\_avg\_students = []

for student, marks in student\_data.items():

avg = sum(marks) / len(marks)

if avg < lowest\_avg:

lowest\_avg = avg

lowest\_avg\_students = [student]

elif avg == lowest\_avg:

lowest\_avg\_students.append(student)

return lowest\_avg\_students

def main():

student\_data = get\_student\_data()

highest\_avg\_students = identify\_highest\_average(student\_data)

highest\_assignment\_students = identify\_highest\_assignment(student\_data)

lowest\_lab\_students = identify\_lowest\_lab(student\_data)

lowest\_avg\_students = identify\_lowest\_average(student\_data)

print(" ".join(highest\_avg\_students))

print(" ".join(highest\_assignment\_students))

print(" ".join(lowest\_lab\_students))

print(" ".join(lowest\_avg\_students))

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

main()

**Ex. No. : 8.4 Date: 28/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Input

REC

[Sample](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) Output

REC is worth 5 points.

**For example:**

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

**def scrabble\_score(word):**

**letter\_to\_points = {1: "AEILNORSTU",2: "DG",3: "BCMP",4: "FHVWY",5: "K",8: "JX",10: "QZ"}**

**points\_dict = {}**

**for points, letters in letter\_to\_points.items():**

**for letter in letters:**

**points\_dict[letter] = points**

**total\_score = 0**

**for letter in word.upper():**

**total\_score += points\_dict.get(letter, 0)**

**print(f"{word} is worth {total\_score} points.")**

**word = input()**

**scrabble\_score(word)**

**Ex. No. : 8.5 Date: 28/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 |

n = int(input())

test\_dict = {}

for \_ in range(n):

data = input().split()

key = data[0]

values = list(map(int, data[1:]))

test\_dict[key] = values

sum\_dict = {key: sum(values) for key, values in test\_dict.items()}

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

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

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

**09 – Dictionary**

**Ex. No. : 9.1 Date: 31/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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(n):**

**sum=0**

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

**if(n%i==0):**

**sum+=i**

**if(sum>n):**

**return("Yes")**

**else:**

**return("No")**

**def main():**

**n=int(input())**

**print(abundant(n))**

**Ex. No. : 9.2 Date: 31/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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):**

**if n <= 0:**

**return "not ugly"**

**for p in [2, 3, 5]:**

**while n % p == 0:**

**n //= p**

**return "ugly" if n == 1 else "not ugly"**

**def main():**

**n=int(input())**

**print(checkUgly(n))**

**Ex. No. : 9.3 Date: 31/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



complete function to implement coin change making problem i.e. finding the minimum

number of coins of certain denominations that add up to given amount of money.

The only available coins are of values 1, 2, 3, 4

Input Format:

Integer input from stdin.

Output Format:

return the minimum number of coins required to meet the given target.

Example Input:

16

Output:

4

Explanation:

We need only 4 coins of value 4 each

Example Input:

25

Output:

7

Explanation:

We need 6 coins of 4 value, and 1 coin of 1 value

**def coinChange(n):**

**coins = [4, 3, 2, 1]**

**count = 0**

**for coin in coins:**

**while n >= coin:**

**n -= coin**

**count += 1**

**return count**

**def main():**

**n=int(input())**

**print(coinChange(n))**

**Ex. No. : 9.4 Date: 31/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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))

**Ex. No. : 9.5 Date: 31/05/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

They are planning to offer a flat discount. The discount value is calculated as the sum of all

the prime digits in the total bill amount.

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

Constraints

1 <= orderValue< 10e100000

Input

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

Output

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

Example Input578

Output

12

**For example:**

def christmasDiscount(n):

prime\_digits = {'2', '3', '5', '7'}

n\_str=str(n)

discount = 0

for char in n\_str:

if char in prime\_digits:

discount += int(char)

return discount

def main():

n=int(input())

print(christmasDiscount(n))

**10 - Searching & Sorting**

**Ex. No. : 10.1 Date: 05/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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")

**Ex. No. : 10.2 Date: 05/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



**A**n 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. : 10.3 Date: 05/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 10.4 Date: 05/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 10.5 Date: 05/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

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)))

**11- Exceptions**

**Ex. No. : 11.1 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 perform\_operations(num1, num2):

try:

division\_result = num1 / num2

modulo\_result = int(num1 % num2)

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

print(f"Modulo result: {modulo\_result}")

except ZeroDivisionError:

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

except ValueError:

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

def main():

try:

num1 = float(input())

num2 = float(input())

perform\_operations(num1, num2)

except ValueError:

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

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

main()

**Ex. No. : 11.2 Date:09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. |

def get\_age():

while True:

try:

age\_input = input("").strip() # Remove leading/trailing whitespace

if not age\_input:

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

break

continue # Continue to the next iteration of the loop

age = int(age\_input)

if age < 0:

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

break

else:

return age

except ValueError:

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

break

except EOFError:

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

break # Exit the loop if EOFError occurs

def main():

age = get\_age()

if age is not None:

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

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

main()

**Ex. No. : 11.3 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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() |

def get\_number\_in\_range(prompt, min\_val, max\_val):

while True:

try:

num = int(input(prompt))

if num < min\_val or num > max\_val:

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

else:

print("Valid input.")

break

except ValueError:

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

except EOFError:

break # Exiting the loop when EOFError occurs

def main():

min\_val = 1

max\_val = 100

get\_number\_in\_range("", min\_val, max\_val)

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

main()

**Ex. No. : 11.4 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Problem Description:

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

Input Format:

User inputs a number.

Output Format:

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

**For example:**

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

import math

def calculate\_square\_root(num):

try:

if num < 0:

print("Error: Cannot calculate the square root of a negative number.")

else:

sqrt\_result = math.sqrt(num)

print(f"The square root of {num} is {sqrt\_result:.2f}")

except ValueError:

print("Error: could not convert string to float")

def main():

try:

num = float(input())

calculate\_square\_root(num)

except ValueError:

print("Error: could not convert string to float")

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

main()

**Ex. No. : 11.5 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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 safe\_division(num1, num2):

try:

result = num1 / num2

print(result)

except ZeroDivisionError:

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

except TypeError:

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

def main():

try:

num1 = float(input())

num2 = float(input())

safe\_division(num1, num2)

except ValueError:

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

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

main()

**12-Modules**

**Ex. No. : 12.1 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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

**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. : 12.2 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 12.3 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 12.4 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



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. : 12.5 Date: 09/06/24**

**Register No.: 230701022 Name: Akshay Venkat Krishna**



Background:

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

Problem Statement:

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

Input Format:

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

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

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

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

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

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

Output Format:

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

from collections import Counter

X = int(input())

N = map(int,input().split())

x = int(input())

L = map(tuple,(map(int,input().split()) for \_ in range(x)))

n = Counter(N)

p =0

for i in L:

if i[0] in n.keys() and n[i[0]] >0 :

n[i[0]] = n[i[0]]-1

p = p+i[1]

print(p)