**RAJALAKSHMI ENGINEERING COLLEGE**

**RAJALAKSHMI NAGAR, THANDALAM – 602 105**



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

# 

2023-2024

2

231501027

1/AIML/A

K.ASWIN

# 



A

AIML

2023-2024

**K.Aswin**

231501027

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.**  **No.** | **Date** | **Title** | **Page**  **No.** | **Teacher’s Signature /**  **Remarks** |
| **Introduction to python-Variables-Datatypes-Input/Output-Formatting** | | | | |
| 1.1 |  | Converting Input Strings |  |  |
| 1.2 |  | Gross salary |  |  |
| 1.3 |  | Square Root |  |  |
| 1.4 |  | Gain percent |  |  |
| 1.5 |  | Deposits |  |  |
| 1.6 |  | Carpenter |  |  |
| **Operators in Python** | | | | |
| 2.1 |  | Widgets and Gizmos |  |  |
| 2.2 |  | Doll Sings |  |  |
| 2.3 |  | Birthday party |  |  |
| 2.4 |  | Hamming Weight |  |  |
| 2.5 |  | Compound Interest |  |  |
| 2.6 |  | Eligible to donate blood |  |  |
| 2.7 |  | C or D |  |  |
| 2.8 |  | Troy Battle |  |  |
| 2.9 |  | Tax and Tip |  |  |
| 2.10 |  | Return last digit of the given number |  |  |
| **Selection Structures in Python** | | | | |
| 3.1 |  | Admission eligibility |  |  |
| 3.2 |  | Classifying triangles |  |  |
| 3.3 |  | Electricity Bill |  |  |
| 3.4 |  | IN/OUT |  |  |
| 3.5 |  | Vowel or Constant |  |  |
| 3.6 |  | Leap Year |  |  |
| 3.7 |  | Month name to Days |  |  |
| 3.8 |  | Pythagorean triple |  |  |
| 3.9 |  | Second Last Digit |  |  |
| 3.10 |  | Chinese Zodiac |  |  |
| **Algorithmic Approach: Iteration Control Structures** | | | | |
| 4.1 |  | Factors of a Number |  |  |
| 4.2 |  | Non-Repeated Digits Count |  |  |
| 4.3 |  | Prime Checking |  |  |
| 4.4 |  | Next Perfect Square |  |  |
| 4.5 |  | Nth Fibonacci |  |  |
| 4.6 |  | Disarium Number |  |  |
| 4.7 |  | Sum of Series |  |  |
| 4.8 |  | Unique Digits Count |  |  |
| 4.9 |  | Product of single digits |  |  |
| 4.10 |  | Perfect Square After adding One |  |  |
| **Strings in Python** | | | | |
| 5.1 |  | Count chars |  |  |
| 5.2 |  | Decompress the String |  |  |
| 5.3 |  | First N Common Characters |  |  |
| 5.4 |  | Remove Characters |  |  |
| 5.5 |  | Remove Palindrome Words |  |  |
| 5.6 |  | Return Second Word in Uppercase |  |  |
| 5.7 |  | Reverse String |  |  |
| 5.8 |  | String characters balance Test |  |  |
| 5.9 |  | Unique Names |  |  |
| 5.10 |  | Username Domain Extension |  |  |
| **List in Python** | | | | |
| 6.1 |  | Monotonic array |  |  |
| 6.2 |  | Check pair with difference k . |  |  |
| 6.3 |  | Count Elements |  |  |
| 6.4 |  | Distinct Elements in an Array |  |  |
| 6.5 |  | Element Insertion |  |  |
| 6.6 |  | Find the Factor |  |  |
| 6.7 |  | Merge list |  |  |
| 6.8 |  | Merge Two Sorted Arrays Without Duplication |  |  |
| 6.9 |  | Print Element Location |  |  |
| 6.10 |  | Strictly increasing |  |  |
| **Tuples & Set** | | | | |
| 7.1 |  | Binary String |  |  |
| 7.2 |  | Check Pair |  |  |
| 7.3 |  | DNA Sequence |  |  |
| 7.4 |  | Print repeated no |  |  |
| 7.5 |  | Remove repeated |  |  |
| 7.6 |  | malfunctioning keyboard |  |  |
| 7.7 |  | American keyboard |  |  |
| **Dictionary** | | | | |
| 8.1 |  | Uncommon Words |  |  |
| 8.2 |  | Sort Dictionary By Values Summation |  |  |
| 8.3 |  | Winner Of Election |  |  |
| 8.4 |  | Student Record |  |  |
| 8.5 |  | Scramble Score |  |  |
| **Functions** | | | | |
| 9.1 |  | Abundant Number |  |  |
| 9.2 |  | Automorphic number or not |  |  |
| 9.3 |  | Check Product of Digits |  |  |
| 9.4 |  | Christmas Discount |  |  |
| 9.5 |  | Coin Change |  |  |
| 9.6 |  | Difference Sum |  |  |
| 9.7 |  | Ugly number |  |  |
| **Searching & Sorting** | | | | |
| 10.1 |  | Merge Sort |  |  |
| 10.2 |  | Bubble Sort |  |  |
| 10.3 |  | Peak Element |  |  |
| 10.4 |  | Binary Search |  |  |
| 10.5 |  | Frequency of Numbers |  |  |

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

**Input/Output-Formatting**

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

**Register No.: 231501027 Name: K.Aswin**

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

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

*Sample Input:*

10

10.9

*Sample Output:* **For example:**

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

10,<class 'int'>

10.9,<class 'float'>

**Program:  
 def convert\_and\_display(s):**

**try:**

**int\_val = int(s)**

**print("Integer value: ", int\_val)**

**print("Type: ", type(int\_val))**

**except ValueError:**

**try:**

**float\_val = float(s)**

**print("Float value: ", float\_val)**

**print("Type: ", type(float\_val))**

**except ValueError:**

**print("Invalid input. Cannot be converted to integer or float.")**

**s = input("Enter a string: ")**

**convert\_and\_display(s)**

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

**Register No.: 231501027 Name:K.Aswin**

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

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

*Sample Input:*

10000

*Sample Output:*

16000

**For example:**

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

*Program:*

*def calculate\_gross\_salary():*

*basic\_salary = float(input("Enter Ramesh's basic salary: "))*

*dearness\_allowance = 0.4 \* basic\_salary*

*house\_rent\_allowance = 0.2 \* basic\_salary*

*gross\_salary = basic\_salary + dearness\_allowance + house\_rent\_allowance*

*print("Ramesh's gross salary is: ", gross\_salary)*

*calculate\_gross\_salary()*

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

**Register No.: 231501027 Name: K.Aswin**

**Square Root**

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

Sample Input:

8.00

Sample Output:

2.828

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

**For example:**

Program:

import math

def find\_square\_root():

num = float(input("Enter a floating point number: "))

square\_root = math.sqrt(num)

print("The square root of", num, "is {:.3f}".format(square\_root))

find\_square\_root()

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

**Register No.: 231501027 Name: K.Aswin**

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

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

Input Format:

The first line contains the Rs X

The second line contains Rs Y

The third line contains Rs Z

Sample Input:

10000

250

15000

Sample Output:

46.34 is the gain percent.

| **Input** | **Result** |
| --- | --- |
| 45500  500  60000 | 30.43 is the gain percent. |

**For example:**

Program:

def find\_gain\_percent():

x = float(input("Enter the cost price of the scooter (Rs.): "))

y = float(input("Enter the repair cost (Rs.): "))

z = float(input("Enter the selling price of the scooter (Rs.): "))

if z <= x + y:

print("Selling price should be greater than the cost price plus repair cost.")

else:

total\_cost = x + y

gain = z - total\_cost

gain\_percent = (gain / total\_cost) \* 100

print("Alfred's gain percent is: {:.2f}%".format(gain\_percent))

find\_gain\_percent()

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

**Register No.: 231501027 Name: K.Aswin**

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

In many jurisdictions, a small deposit is added to drink containers to encourage people to recycle them. In one particular jurisdiction, drink containers holding one liter or less have a $0.10 deposit and drink containers holding more than one liter have a $0.25 deposit. Write a program that reads the number of containers of each

size(less and more)  from the user. Your program should continue by computing and displaying the refund that will be received for returning those containers. Format the output so that it includes a dollar sign and always displays exactly two decimal places.

Sample Input

10

20

Sample Output

Your total refund will be $6.00.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 20  20 | Your total refund will be $7.00. |

**Program:  
 def calculate\_refund():**

**containers\_less\_than\_one\_liter = int(input("Enter the number of containers holding one liter or less: "))**

**containers\_more\_than\_one\_liter = int(input("Enter the number of containers holding more than one liter: "))**

**deposit\_less\_than\_one\_liter = 0.10**

**deposit\_more\_than\_one\_liter = 0.25**

**refund\_less\_than\_one\_liter = containers\_less\_than\_one\_liter \* deposit\_less\_than\_one\_liter**

**refund\_more\_than\_one\_liter = containers\_more\_than\_one\_liter \* deposit\_more\_than\_one\_liter**

**total\_refund = refund\_less\_than\_one\_liter + refund\_more\_than\_one\_liter**

**print("The refund for returning the containers is: ${:.2f}".format(total\_refund))**

**calculate\_refund()**

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

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

**Hint:**

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

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

number = -20

absolute\_number = abs(number)

print(absolute\_number)

# Output: 2

**Sample Input:**

450

**Sample Output:**

weekdays 10.38

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

weekend 0.38

**For example:**

Program:

total\_salary = float(input("Enter the total salary paid to Justin: "))

total\_salary = (x + 10) \* 50 + x \* 80

x = (total\_salary - 500) / 130

weekend\_hours = x

weekday\_hours = x + 10

print("Justin worked", int(weekend\_hours), "hours on weekends.")

print("Justin worked", int(weekday\_hours), "hours on weekdays.")

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

Sample Input

10

20

Sample Output

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

**For example:**

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

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

**Register No.: 231501027 Name: K.Aswin**

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

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

**Program:**

**def calculate\_total\_weight(widgets, gizmos):**

**widget\_weight = 75 # grams per widget**

**gizmo\_weight = 112 # grams per gizmo**

**total\_weight = (widgets \* widget\_weight) + (gizmos \* gizmo\_weight)**

**return total\_weight**

**def main():**

**widgets = int(input("Enter the number of widgets: "))**

**gizmos = int(input("Enter the number of gizmos: "))**

**total\_weight = calculate\_total\_weight(widgets, gizmos)**

**print("Total weight of the parts: {} grams".format(total\_weight))**

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

**main()**

Sample Input

10

Sample Output

True

Explanation:

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

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

**Register No.: 231501027 Name: K.Aswin**

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

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

 IF Lokpaul wins print true, otherwise false.

**Program:**

**def is\_winning\_doll(number):**

**if 1 <= number <= 100 and number % 2 == 0 and number != 0:**

**return True**

**else:**

**return False**

**pressed\_number = int(input("Enter the number pressed: "))**

**print(is\_winning\_doll(pressed\_number))**

Input Given:

N-No of friends

P1,P2,P3 AND P4-No of chocolates

OUTPUT:

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

SAMPLE INPUT AND OUTPUT:

5

25

12

10

9

OUTPUT

True False True False

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

**Register No.: 231501027 Name: K.Aswin**

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

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

Program:

def find\_packet(num\_friends, packets):

for packet in packets:

if packet % num\_friends == 0:

return packet

return -1

def main():

num\_friends = int(input("Enter the number of friends: "))

packets = [int(x) for x in input("Enter the number of chocolates in each packet separated by space: ").split()]

packet = find\_packet(num\_friends, packets)

if packet != -1:

print("Buy the packet with {} chocolates, which can be distributed equally among all friends.".format(packet))

else:

print("No packet found that can be distributed equally among all friends.")

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

main()

Sample Input

3

Sample Output:

2

Explanation:

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

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

**Register No.: 231501027 Name:K.Aswin**

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

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

Program:

def count\_ones\_in\_binary(num):

count = 0

while num:

count += num & 1

num >>= 1

return count

def main():

num = int(input("Enter an integer between 0 and 15: "))

if 0 <= num <= 15:

print("Number of '1's in binary form:", count\_ones\_in\_binary(num))

else:

print("Invalid input. Please enter an integer between 0 and 15.")

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

main()

Sample Input:

10000

Sample Output:

Balance as of end of Year 1: $10400.00.

Balance as of end of Year 2: $10816.00.

Balance as of end of Year 3: $11248.64

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

**Register No.: 231501027 Name: K.Aswin**

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

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

.

**Program:**

**def calculate\_balance(deposit, years):**

**interest\_rate = 0.04**

**balance = deposit \* (1 + interest\_rate) \*\* years**

**return balance**

**def main():**

**deposit = float(input("Enter the amount of money deposited into the account: $"))**

**for year in range(1, 4):**

**balance = calculate\_balance(deposit, year)**

**print("After {} year(s), the balance is: ${:.2f}".format(year, balance))**

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

**main()**

 Input Format:

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

 Output Format:

 Display True(IF ELIGIBLE)

Display False (if not eligible)

Sample Input

19

45

Sample Output

True

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

**Register No.:231501027 Name: K.Aswin**

**Eligible to donate blood**

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

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

**Program:**

**def check\_eligibility(age, weight):**

**if age >= 18 and weight > 40:**

**return True**

**else:**

**return False**

**# Test cases**

**age = int(input("Enter your age: "))**

**weight = int(input("Enter your weight (in kg): "))**

**if check\_eligibility(age, weight):**

**print("You are eligible to donate blood. Welcome!")**

**else:**

**print("Sorry, you are not eligible to donate blood.")**

**Input Format:**

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

**Output Format:**

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

**Input 1:**

0

**Output 1:**

C

**Input 2:**

1

**Output 1:**

D

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

**Register No.: 231501027 Name: K.Aswin**

**C or D**

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

Hint:

Use ASCII values of C and D.

**Program:  
 def display\_char(input\_value):**

**return 'C' if not input\_value else 'D'**

**print(display\_char(0))**

**print(display\_char(1))**

**Input format:**

Line 1 has the total number of weapons

Line 2 has the total number of Soldiers.

**Output  Format:**

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

Sample Input:

32

43

Sample Output:'

False

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

**Register No.: 231501027 Name: K.Aswin**

**Troy Battle**

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

Program:

def can\_win\_battle(weapons, soldiers):

if weapons % 3 == 0 and soldiers % 2 == 0:

return True

else:

return False

weapons\_count = 27

soldiers\_count = 6

print(can\_win\_battle(weapons\_count, soldiers\_count)) # Output will be True

Sample Input

100

Sample Output

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

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

**Register No.: 231501027 Name: K.Aswin**

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

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

Program:

def calculate\_total\_cost(meal\_cost):

tax\_rate = 0.05

tip\_rate = 0.18

tax\_amount = meal\_cost \* tax\_rate

tip\_amount = meal\_cost \* tip\_rate

total\_cost = meal\_cost + tax\_amount + tip\_amount

return tax\_amount, tip\_amount, total\_cost

def main():

meal\_cost = float(input("Enter the cost of the meal: $"))

tax\_amount, tip\_amount, total\_cost = calculate\_total\_cost(meal\_cost)

print("Tax: ${:.2f}".format(tax\_amount))

print("Tip: ${:.2f}".format(tip\_amount))

print("Total cost: ${:.2f}".format(total\_cost))

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

main()

**For example:**

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

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

**Register No.: 231501027 Name: K.Aswin**

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

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

The last digit should be returned as a positive number.

For example,

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

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

Program:

def last\_digit(n):

return abs(n) % 10

n = int(input("Enter a number: "))

result = last\_digit(n)

print("The last digit of the number is:", result)

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

**For example:**

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

**Register No.: 231501027 Name: K.Aswin**

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

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

Marks in Maths >= 65

Marks in Physics >= 55

Marks in Chemistry >= 50

Or

| **Input** | **Result** |
| --- | --- |
| 50  80  80 | The candidate is eligible |

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

Program:

maths\_marks = float(input("Enter the marks in Maths: "))

physics\_marks = float(input("Enter the marks in Physics: "))

chemistry\_marks = float(input("Enter the marks in Chemistry: "))

total\_marks = maths\_marks + physics\_marks + chemistry\_marks

if (maths\_marks >= 65 and physics\_marks >= 55 and chemistry\_marks >= 50) or total\_marks >= 180:

print("You are eligible for admission to the professional course.")

else:

print("You are not eligible for admission to the professional course.")

**For example:**

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

**Register No.: 231501027 Name: K.Aswin**

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

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

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

| **Input** | **Result** |
| --- | --- |
| 40  40  80 | That's a isosceles triangle |

Sample Input 1

60

60

60

Sample Output 1

That's a equilateral triangle

**Program:**

**side1 = float(input("Enter the length of side 1: "))**

**side2 = float(input("Enter the length of side 2: "))**

**side3 = float(input("Enter the length of side 3: "))**

**if side1 == side2 == side3:**

**triangle\_type = "equilateral"**

**elif side1 == side2 or side2 == side3 or side1 == side3:**

**triangle\_type = "isosceles"**

**else:**

**triangle\_type = "scalene"**

**print("The triangle is", triangle\_type)**

**For example:**

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

**Register No.: 231501027 Name: K.Aswin**

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

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

| **Input** | **Result** |
| --- | --- |
| 500 | 1035.00 |

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

**Program:**

**units = int(input("Enter the units consumed: "))**

**if units <= 199:**

**charge = units \* 1.20**

**elif 200 <= units < 400:**

**charge = units \* 1.50**

**elif 400 <= units < 600:**

**charge = units \* 1.80**

**else:**

**charge = units \* 2.00**

**if charge > 400:**

**surcharge = charge \* 0.15**

**total\_amount = charge + surcharge**

**else:**

**total\_amount = max(charge, 100)**

**print("Total amount to be paid: Rs.", round(total\_amount, 2))**

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

**Register No.: 231501027 Name: K.Aswin**

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

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

 Input Format:

Input consists of 2 integers.

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

 Output Format:

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

 Sample Input and Output:

| **Input** | **Result** |
| --- | --- |
| 8  3 | OUT |

 Input

 8

 3

 Output

 OUT

Program:

total\_problems = int(input("Enter the total number of problems given last week: "))

problems\_solved = int(input("Enter the number of problems solved by the student: "))

min\_required = total\_problems // 2

if problems\_solved >= min\_required:

print("You are allowed to enter the lab.")

else:

print("You are not allowed to enter the lab.")

**For example:**

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

**Register No.: 231501027 Name: K.Aswin**

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

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

Sample Input 1

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

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.

**Program:**

**letter = input("Enter a letter: ").lower()**

**if letter in "aeiou":**

**print(f"{letter} is a vowel.")**

**elif letter == "y":**

**print(f"{letter} is sometimes a vowel and sometimes a consonant.")**

**else:**

**print(f"{letter} is a consonant.")**

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

**Register No.: 231501027 Name: K.Aswin**

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

Most years have 365 days. However, the time required for the Earth to orbit the Sun is actually slightly more than that. As a result, an [extra](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5753) day, February 29, is included in some years to correct for this difference. Such years are referred to as leap years. The rules for determining whether or not a year is a leap year follow:

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

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

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

• All other years are not leap years.

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

Sample Input 1

1900

Sample Output 1

1900 is not a leap year.

Sample Input 2

2000

Sample Output 2

2000 is a leap year.

Program:

year = int(input("Enter a year: "))

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

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

else:

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

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

**Register No.: 231501027 Name: K.Aswin**

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

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

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.

Program:

def get\_number\_of\_days(month):

if month.lower() == "february":

return "28 or 29 days"

elif month.lower() in ["april", "june", "september", "november"]:

return "30 days"

else:

return "31 days"

month = input("Enter the name of a month: ")

print("Number of days in", month, ":", get\_number\_of\_days(month))

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

**Register No.: 231501027 Name: K.Aswin**

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

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

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

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

**Sample Input**

3

5

4

**Sample Output**

Yes

Program:  
 def is\_pythagorean\_triple(a, b, c):

return a\*\*2 + b\*\*2 == c\*\*2 or a\*\*2 + c\*\*2 == b\*\*2 or b\*\*2 + c\*\*2 == a\*\*2

a = int(input("Enter the first number: "))

b = int(input("Enter the second number: "))

c = int(input("Enter the third number: "))

if is\_pythagorean\_triple(a, b, c):

print("The numbers", a, b, "and", c, "form a Pythagorean triple.")

else:

print("The numbers", a, b, "and", c, "do not form a Pythagorean triple.")

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

**Register No.: 231501027 Name: K.Aswin**

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

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

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

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

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

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

**Program:**

def second\_last\_digit(n):

return (n // 10) % 10

num = int(input("Enter a number: "))

result = second\_last\_digit(num)

print("The second last digit of the number", num, "is", result)

.

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

**Register No.: 231501027 Name: K.Aswin**

**Chinese Zodiac**

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

Year Animal

2000 Dragon

2001 Snake

2002 Horse

2003 Sheep

2004 Monkey

2005 Rooster

2006 Dog

2007 Pig

2008 Rat

2009 Ox

2010 Tiger

2011 Hare

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

Sample Input 1

2010

Sample Output 1

2010 is the year of the Tiger.

Sample Input 2

2020

Sample Output 2

2020 is the year of the Rat

Program:  
 def get\_chinese\_zodiac\_animal(year):

animals = ["Monkey", "Rooster", "Dog", "Pig", "Rat", "Ox", "Tiger", "Rabbit", "Dragon", "Snake", "Horse", "Sheep"]

return animals[year % 12]

year = int(input("Enter a year: "))

animal = get\_chinese\_zodiac\_animal(year)

print("The animal associated with the year", year, "is the", animal)

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

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

**Register No.: 231501027 Name: K.Aswin**

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

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

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

**For example:**

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

Program:  
 def find\_factors(n):

factors = []

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

if n % i == 0:

factors.append(i)

return factors

num = int(input("Enter a number: "))

factors = find\_factors(num)

print("The factors of", num, "are:", factors)

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

**Register No.: 231501027 Name: K.Aswin**

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

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

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

Some examples are as below.

If the given number is 292, the program should return 1 because there is only 1 non-­repeated digit '9' in this number

If the given number is 1015, the program should return 2 because there are 2 non-repeated digits in this number, '0', and '5'.

If the given number is 108, the program should return 3 because there are 3 non-­repeated digits in this number, '1', '0', and '8'.

If the given number is 22, the function should return 0 because there are NO non-­repeated digits in this number.

**Program:  
 def count\_non\_repeated\_digits(n):**

**digit\_count = {}**

**for digit in str(n):**

**if digit in digit\_count:**

**digit\_count[digit] += 1**

**else:**

**digit\_count[digit] = 1**

**non\_repeated\_count = 0**

**for count in digit\_count.values():**

**if count == 1:**

**non\_repeated\_count += 1**

**return non\_repeated\_count**

**n = int(input("Enter a number: "))**

**result = count\_non\_repeated\_digits(n)**

**print("The count of non-repeated digits in", n, "is:", result)**

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

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| 7 | 2 |
| 10 | 1 |

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

**Register No.: 231501027 Name:K.Aswin**

**Prime Checking**

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

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

**Program:**

**def is\_prime(n):**

**if n <= 1:**

**return 1**

**if n == 2:**

**return 2**

**if n % 2 == 0:**

**return 1**

**max\_divisor = int(n\*\*0.5) + 1**

**for d in range(3, max\_divisor, 2):**

**if n % d == 0:**

**return 1**

**return 2**

**n = int(input("Enter a number: "))**

**result = is\_prime(n)**

**print("The number", n, "is prime?" , result == 2)**

Input Format:

Integer input from stdin.

Output Format:

Perfect square greater than N.

Example Input:

10

Output:

16

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

**Register No.: 231501027 Name:K.Aswin**

**Next Perfect Square**

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

**Program:**

**import math**

**def next\_perfect\_square(n):**

**root = math.ceil(math.sqrt(n))**

**return root \* root**

**n = int(input("Enter a number: "))**

**result = next\_perfect\_square(n)**

**print("The next perfect square greater than", n, "is:", result)**

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

**Register No.: 231501027 Name: K.Aswin**

**Nth Fibonacci**

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

NOTE: Fibonacci series looks like –

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

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

• first Fibonacci number is 0,

• second Fibonacci number is 1,

• third Fibonacci number is 1,

• fourth Fibonacci number is 2,

• fifth Fibonacci number is 3,

• sixth Fibonacci number is 5,

• seventh Fibonacci number is 8, and so on.

**For example:**

**Input:**

**7**

**Output**

**8**

**Program:**

**def fibonacci(n):**

**if n <= 0:**

**return "Input should be a positive integer"**

**elif n == 1:**

**return 0**

**elif n == 2:**

**return 1**

**else:**

**a, b = 0, 1**

**for \_ in range(2, n):**

**a, b = b, a + b**

**return b**

**n = int(input("Enter a positive integer: "))**

**result = fibonacci(n)**

**print("The {}th number in the Fibonacci series is: {}".format(n, result))**

Input Format:

Single Integer Input from stdin.

Output Format:

Yes or No.

Example Input:

175

Output:

Yes

Explanation

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

Example Input:

123

Output:

No

**For example:**

| **Input** | **Result** |
| --- | --- |
| 175 | Yes |
| 123 | No |

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

**Register No.: 231501027 Name: K.Aswin**

**Disarium Number**

A Number is said to be Disarium number when the sum of its digit raised to the power of their respective positions becomes equal to the number itself. Write a [program](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) to print number is Disarium or not.

**Program:**

**def is\_disarium(n):**

**num\_str = str(n)**

**num\_len = len(num\_str)**

**sum\_of\_powers = 0**

**for i in range(num\_len):**

**digit = int(num\_str[i])**

**sum\_of\_powers += digit \*\* (i + 1)**

**return sum\_of\_powers == n**

**n = int(input("Enter a number: "))**

**if is\_disarium(n):**

**print(f"{n} is a Disarium number.")**

**else:**

**print(f"{n} is not a Disarium number.")**

Sample Test Cases

Test Case 1

Input

4

Output

1234

Explanation:

as input is 4, have to take 4 terms.

1 + 11 + 111 + 1111

Test Case 2

Input

6

Output

123456

**For example:**

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

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

**Register No.: 231501027 Name: K.Aswin**

**Sum of Series**

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

**Program:**

**def sum\_of\_series(n):**

**total\_sum = 0**

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

**num\_str = '1' \* i**

**total\_sum += int(num\_str)**

**return total\_sum**

**n = int(input("Enter the number of terms: "))**

**result = sum\_of\_series(n)**

**print("The sum of the series is:", result)**

**For example:**

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

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

**Register No.: 231501027 Name: K.Aswin**

**Unique Digit Count**

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

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

For e.g.

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

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

**Program:**

**def count\_unique\_digits(n):**

**unique\_digits = set()**

**while n > 0:**

**digit = n % 10**

**unique\_digits.add(digit)**

**n //= 10**

**return len(unique\_digits)**

**n = int(input("Enter a number: "))**

**result = count\_unique\_digits(n)**

**print("The count of unique digits is:", result)**

Input Format:

Single Integer input.

Output Format:

Output displays Yes if condition satisfies else prints No.

Example Input:

14

Output:

Yes

Example Input:

13

Output:

No

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

**Register No.: 231501027 Name: K.Aswin**

**Product of single digit**

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

**Program:**

**def can\_be\_represented(n):**

**i = 2**

**factors = []**

**while i \* i <= n:**

**if n % i:**

**i += 1**

**else:**

**n //= i**

**factors.append(i)**

**if n > 1:**

**factors.append(n)**

**single\_digit\_factors = [x for x in factors if 0 < x < 10]**

**product = 1**

**for digit in single\_digit\_factors:**

**product \*= digit**

**return product == n**

**n = int(input("Enter a number: "))**

**result = can\_be\_represented(n)**

**print("The number can be represented as a product of single digit numbers:" if result else "The number cannot be represented as a product of single digit numbers:")**

Input Format:

Single integer input.

Output Format:

Yes or No.

Example Input:

24

Output:

Yes

Example Input:

26

Output:

No

**For example:**

| **Input** | **Result** |
| --- | --- |
| 24 | Yes |

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

**Register No.: 231501027 Name: K.Aswin**

**Perfect Square After adding One**

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

Program:

import math

def can\_be\_perfect\_square(n):

num = n + 1

root = math.sqrt(num)

return int(root + 0.5) \*\* 2 == num

n = int(input("Enter a number: "))

result = can\_be\_perfect\_square(n)

print("The number can be made a perfect square after adding 1:" if result else "The number cannot be made a perfect square after adding 1:")

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

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

**Register No.: 231501027 Name: K.Aswin**

**Balanced Array**

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

Example

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

·         the sum of the first three elements, 1+2+3=6. The value of the last element is 6.

·         Using zero based indexing, arr[3]=4 is the pivot between the two subarrays.

·         The index of the pivot is 3.

Constraints

·         3 ≤ n ≤ 105

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

·         It is guaranteed that a solution always exists.

The first line contains an integer n, the size of the array arr.

Each of the next n lines contains an integer, arr[i], where 0 ≤ i < n.

Sample Case 0

Sample Input 0

4

1

2

3

3

Sample Output 0

2

Explanation 0

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

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

·         The index of the pivot is 2.

Sample Case 1

Sample Input 1

3

1

2

1

Sample Output 1

1

Explanation 1

·         The first and last elements are equal to 1.

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

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

·         The index of the pivot is 1.

**For example:**

**Program:  
 def find\_pivot\_index(arr):**

**total\_sum = sum(arr)**

**left\_sum = 0**

**for i, num in enumerate(arr):**

**total\_sum -= num**

**if left\_sum == total\_sum:**

**return i**

**left\_sum += num**

**return -1**

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

**print(find\_pivot\_index(arr)) # Output: 3**

**arr = [1, 2, 3, 4, 5, 6]**

**print(find\_pivot\_index(arr)) # Output: -1**

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

**Register No.: 231501027 Name: K.Aswin**

**Check pair with difference k**

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

Input Format

1.      First line is number of test cases T. Following T lines contain:

2.      N, followed by N integers of the array

3.      The non-negative integer k

Output format

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

Input

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

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

**For example:**

**Program:**

**def find\_indices(A, k):**

**i, j = 0, len(A) - 1**

**while i < j:**

**diff = A[i] - A[j]**

**if diff == k:**

**return True**

**elif diff < k:**

**i += 1**

**else:**

**j -= 1**

**return False**

**A = [1, 3, 5, 7, 9]**

**k = 2**

**print(find\_indices(A, k)) # Output: True**

**A = [1, 3, 5, 7, 9]**

**k = 4**

**print(find\_indices(A, k)) # Output: False**

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

**Register No.: 231501027 Name: K.Aswin**

**Count Elements**

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

Sample Test Cases

 Test Case 1

 Input

 7

23

45

23

56

45

23

40

 Output

 23 occurs 3 times

45 occurs 2 times

56 occurs 1 times

40 occurs 1 times

Program:

 def count\_frequency(arr):

frequency\_dict = {}

for element in arr:

if element in frequency\_dict:

frequency\_dict[element] += 1

else:

frequency\_dict[element] = 1

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

print(f"Frequency of {key} is {value}")

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

print("Frequency of each element in the array is:")

count\_frequency(arr)

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

**Register No.: 231501027 Name: K.Aswin**

**Distinct Elements in an Array**

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

Input Format:

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

Second line take n Integers which is inputs of array.

Output Format:

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

Example Input:

5

1

2

2

3

4

Output:

1 2 3 4

Example Input:

6

1

1

2

2

3

3

Output:

1 2 3

For example:

Input Result

5

1

2

2

3

4

1 2 3 4

6

1

1

2

2

3

3

1 2 3

**Program:  
 def print\_distinct\_elements(arr):**

**distinct\_elements = set(arr)**

**for element in distinct\_elements:**

**print(element)**

**arr = [1, 2, 3, 4, 2, 3, 5, 6, 7, 8, 9, 5, 6]**

**print("Distinct elements in the array are:")**

**print\_distinct\_elements(arr)**

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

**Register No.: 231501027 Name: K.Aswin**

**Element Insertion**

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

Sample Test Cases

Test Case 1

Input

1

3

4

5

6

7

8

9

10

11

2

Output

ITEM to be inserted:2

After insertion array is:

1

2

3

4

5

6

7

8

9

10

11

Test Case 2

Input

11

22

33

55

66

77

88

99

110

120

44

Output

ITEM to be inserted:44

After insertion array is:

11

22

33

44

55

66

77

88

99

110

120

**Program:  
 def insert\_into\_sorted\_array(arr, new\_element):**

**new\_arr = [0] \* (len(arr) + 1)**

**j = 0**

**for i in range(len(arr)):**

**element, insert it**

**if new\_element < arr[i]:**

**new\_arr[j] = new\_element**

**j += 1**

**while i < len(arr):**

**new\_arr[j] = arr[i]**

**i += 1**

**j += 1**

**break**

**else:**

**new\_arr[j] = arr[i]**

**j += 1**

**array, append it**

**if j == len(arr):**

**new\_arr[j] = new\_element**

**return new\_arr**

**arr = [1, 3, 5, 7, 9]**

**new\_element = 6**

**result = insert\_into\_sorted\_array(arr, new\_element)**

**print("The new array is:", result)**

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

**Register No.: 231501027 Name: K.Aswin**

**Find the Factor**

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the pth element of the [list](http://118.185.187.137/moodle/mod/resource/view.php?id=732), sorted ascending. If there is no pth element, return 0.

**Constraints**

1 ≤ n ≤ 1015

1 ≤ p ≤ 109

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

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

**Sample Case 0**

**Sample Input 0**

10

3

**Sample Output 0**

5

**Explanation 0**

Factoring n = 10 results in {1, 2, 5, 10}. Return the p = 3rd factor, 5, as the answer.

**Sample Case 1**

**Sample Input 1**

10

5

**Sample Output 1**

0

**Explanation 1**

Factoring n = 10 results in {1, 2, 5, 10}. There are only 4 factors and p = 5, therefore 0 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.

| **Input** | **Result** |
| --- | --- |
| 10  3 | 5 |
| 10  5 | 0 |
| 1  1 | 1 |

**For example:**

**Program:**

**def find\_pth\_factor(n, p):**

**factors = [i for i in range(1, n + 1) if n % i == 0]**

**factors.sort()**

**if p > len(factors):**

**return 0**

**else:**

**return factors[p - 1]**

**n = int(input("Enter a number: "))**

**p = int(input("Enter the position: "))**

**result = find\_pth\_factor(n, p)**

**print("The {}th factor of {} is {}".format(p, n, result))**

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

**Register No.: 231501027 Name: K.Aswin**

**Merge List**

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

Input:

m : row size

n: column size

list1 and list 2 : Two lists

Output

Zipped List : List which combined both list1 and list2

Sample test case

Sample input

2

2

1

3

5

7

2

4

6

8

Sample Output

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

**Program:**

**def zip\_lists(list1, list2):**

**return [list(zip(a, b)) for a, b in zip(list1, list2)]**

**list1 = [[1, 2], [3, 4], [5, 6]]**

**list2 = [[7, 8], [9, 10], [11, 12]]**

**result = zip\_lists(list1, list2)**

**print("The zipped lists are:", result)**

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

**Register No.: 231501027 Name: K.Aswin**

**Merge Two Sorted Arrays Without Duplication**

Output is a merged array without duplicates.

Input Format

N1 - no of elements in array 1

Array elements for array 1

N2 - no of elements in array 2

Array elements for array2

Output Format

Display the merged array

Sample Input 1

5

1

2

3

6

9

4

2

4

5

10

Sample Output 1

1 2 3 4 5 6 9 10

**Program:**

**def merge\_arrays(arr1, arr2):**

**return list(set(arr1 + arr2))**

**arr1 = input("Enter the first array (separated by spaces): ")**

**arr1 = [x for x in arr1.split()]**

**arr2 = input("Enter the second array (separated by spaces): ")**

**arr2 = [x for x in arr2.split()]**

**result = merge\_arrays(arr1, arr2)**

**print("The merged array without duplicates is:", result)**

.

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

**Register No.: 231501027 Name: K.Aswin**

**Print Element Location**

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

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

5

6

5

7

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

5 is present at location 1

5 is present at location 3

5 is present 2 times in the array.

Sample Test Cases

Test Case 1

Input

4

5

6

5

7

5

Output

5 is present at location 1.

5 is present at location 3.

5 is present 2 times in the array.

Test Case 2

Input

5

67

80

45

97

100

50

Output

50 is not present in the array

Program:

def find\_element(lst, elem):

locations = [i+1 for i, x in enumerate(lst) if x == elem]

count = len(locations)

return locations, count

lst = input("Enter a list of elements (separated by spaces): ")

lst = [x for x in lst.split()]

elem = input("Enter the element to search for: ")

locations, count = find\_element(lst, elem)

if count == 0:

print("The element is not found in the list.")

else:

print("The element is found at locations:", locations)

print("The element occurs", count, "times in the list.")

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

**Register No.: 2315101027 Name:K.Aswin**

**Strictly increasing**

Write a Python program to check if a given list is strictly increasing or not. Moreover, If removing only one element from the list results in a strictly increasing list, we still consider the list true

Input:

n : Number of elements

List1: List of values

Output

Print "True" if list is strictly increasing or decreasing else print "False"

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

Program:  
 def is\_strictly\_increasing(lst):

if all(lst[i] < lst[i+1] for i in range(len(lst)-1)):

return True

for i in range(len(lst)):

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

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

return True

return False

lst = input("Enter a list of numbers (separated by spaces): ")

lst = [int(x) for x in lst.split()]

if is\_strictly\_increasing(lst):

print("The list is strictly increasing or can be made strictly increasing by removing one element.")

else:

print("The list is not strictly increasing and cannot be made strictly increasing by removing one element.")

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

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

**Register No.: 231501027 Name:K.Aswin**

**Count Chars**

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

For example:

Input Result

rec@123

3

3

1

**Program:**

**def count\_chars(s):**

**letter\_count = 0**

**digit\_count = 0**

**special\_symbol\_count = 0**

**for char in s:**

**if char.isalpha():**

**letter\_count += 1**

**elif char.isdigit():**

**digit\_count += 1**

**else:**

**special\_symbol\_count += 1**

**return letter\_count, digit\_count, special\_symbol\_count**

**s = input("Enter a string: ")**

**letter\_count, digit\_count, special\_symbol\_count = count\_chars(s)**

**print("Letter count:", letter\_count)**

**print("Digit count:", digit\_count)**

**print("Special symbol count:", special\_symbol\_count)**

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

**Register No.: 231501027 Name: K.Aswin**

**Decompress the String**

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

Sample Input 1

a2b4c6

Sample Output 1

aabbbbcccccc

**Program:  
 def common\_chars(s1, s2, n):**

**s2\_set = set(s2)**

**result = []**

**for char in s1:**

**if char in s2\_set:**

**result.append(char)**

**if len(result) == n:**

**break**

**return result**

**s1 = input("Enter first string: ")**

**s2 = input("Enter second string: ")**

**n = int(input("Enter the number of characters: "))**

**print("The common characters are:")**

**for char in common\_chars(s1, s2, n):**

**print(char)**

.

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

**Register No.: 231501027 Name: K.Aswin**

**First N Common Chars**

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

Input Format:

The first line contains S1.

The second line contains S2.

The third line contains N.

Output Format:

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

Boundary Conditions:

2 <= N <= 10

2 <= Length of S1, S2 <= 1000

Example Input/Output 1:

Input:

abcbde

cdefghbb

3

Output:

bcd

Note:

b occurs twice in common but must be printed only once**Program:**

**def common\_chars(s1, s2, n):**

**common\_chars = list(set(s1) & set(s2))**

**common\_chars.sort()**

**return common\_chars[:n]**

**s1 = input("Enter first string: ")**

**s2 = input("Enter second string: ")**

**n = int(input("Enter the number of characters: "))**

**print("The common characters are:")**

**for char in common\_chars(s1, s2, n):**

**print(char)**

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

**Register No.: 231501027 Name:K.Aswin**

**Remove Characters**

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

Constraints

1<= string length <= 200

Sample Input 1

experience

enc

Sample Output 1

xpri

For example:



**Program:  
 def remove\_chars(s1, s2):**

**for char in s2:**

**s1 = s1.replace(char, '')**

**return s1**

**s1 = input("Enter the first string: ")**

**s2 = input("Enter the second string: ")**

**print(remove\_chars(s1, s2))**

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

**Register No.: 231501027 Name:K.Aswin**

**Remove Palindrome Words**

String should contain only the words are not palindrome.

Sample Input 1

Malayalam is my mother tongue

Sample Output 1

is my mother tongue

**Program:**

**def non\_palindrome\_words(sentence):**

**words = sentence.split()**

**non\_palindromes = [word for word in words if word != word[::-1]]**

**return ' '.join(non\_palindromes)**

**sentence = input("Enter a sentence: ")**

**print(non\_palindrome\_words(sentence))**

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

**Register No.: 231501027 Name:K.Aswin**

**Return Second World in Uppercase**

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

For example:

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

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

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

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

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

For example:

Input Result

Wipro Technologies Bangalore

TECHNOLOGIES

Hello World

WORLD

Hello

LESS

**Program:  
 def second\_word\_uppercase(sentence):**

**words = sentence.split()**

**if len(words) < 2:**

**return "The sentence should have at least two words."**

**else:**

**words[1] = words[1].upper()**

**return '.join(words)**

**sentence = input("Enter a sentence: ")**

**print(second\_word\_uppercase(sentence))**

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

**Register No.: 231501027 Name: K.Aswin**

**Revers String**

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

Input:

A&B

Output:

B&A

Explanation: As we ignore '&' and

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

For example:

Input Result

A&x#

x&A#

**Program:**

**def reverse\_string(s):**

**s = list(s)**

**left, right = 0, len(s) - 1**

**while left < right:**

**if not s[left].isalpha():**

**left += 1**

**elif not s[right].isalpha():**

**right -= 1**

**else:**

**s[left], s[right] = s[right], s[left]**

**left, right = left + 1, right - 1**

**return ''.join(s)**

**s = input("Enter a string: ")**

**print(reverse\_string(s))**

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

**Register No.: 231501027 Name: K.Aswin**

**String characters balance Test**

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

For example:

Input Result

Yn

PYnative

True

**Program:  
 def check\_balance(s1, s2):**

**for char in s1:**

**if char not in s2:**

**return "false"**

**return "true"**

**s1 = input("Enter the first string: ")**

**s2 = input("Enter the second string: ")**

**print(check\_balance(s1, s2))**

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

**Register No.: 231501027 Name: K.Aswin**

**Unique Names**

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

**Input:**

first

second

first

third

second

then your program should display:

**Output:**

first

second

third

**Program:  
 def unique\_words():**

**words = set()**

**unique\_words\_list = []**

**while True:**

**word = input("Enter a word (or blank line to finish): ")**

**if word == "":**

**break**

**if word not in words:**

**words.add(word)**

**unique\_words\_list.append(word)**

**for word in unique\_words\_list:**

**print(word)**

**unique\_words()**

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

**Register No.: 231501027 Name: K.Aswin**

**Username Domain Extension**

Given a string S which is of the format USERNAME@DOMAIN.EXTENSION, the program must print the EXTENSION, DOMAIN, USERNAME in the reverse order.

**Input Format**:

The first line contains S.

**Output Format**:

The first line contains EXTENSION.

The second line contains DOMAIN.

The third line contains USERNAME.

**Boundary Condition:**

1 <= Length of S <= 100

Example Input/Output 1:

**Input**:

vijayakumar.r@rajalakshmi.edu.in

**Output**:

edu.in

rajalakshmi

vijayakumar

Program:  
 def reverse\_order(s):

username, domain\_extension = s.split('@')

domain, extension = domain\_extension.split('.')

print(extension, domain, username)

s = input("Enter the string: ")

reverse\_order(s)

**07 - Functions**

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

**Register No.: 231501027 Name: K.Aswin**

**Abundant Number**

An abundant number is a number for which the sum of its proper divisors is greater than the number itself. Proper divisors of the number are those that are strictly lesser than the number.

**Input Format**:

Take input an integer from stdin

**Output Format:**

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

**Example input:**

12

**Output**:

Yes

Explanation

The proper divisors of 12 are: 1, 2, 3, 4, 6, whose sum is 1 + 2 + 3 + 4 + 6 = 16. Since sum of proper divisors is greater than the given number, 12 is an abundant number.

**Example input:**

13

**Output**:

No

**Explanation**

The proper divisors of 13 is: 1, whose sum is 1. Since sum of proper divisors is not greater than the given number, 13 is not an abundant number.

For example:

Test Result

print(abundant(12)) Yes

print(abundant(13)) No

Program:  
 def check\_abundant(n):

if n < 1:

return "Invalid input"

sum\_divisors = 0

for i in range(1, n):

if n % i == 0:

sum\_divisors += i

if sum\_divisors > n:

return "Abundant"

else:

return "Not Abundant"

n = int(input("Enter a number: "))

print(check\_abundant(n))

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

**Register No.: 231501027 Name: K.Aswin**

**Automorphic number or not**

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

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

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

Input Format:

Take a Integer from Stdin

Output Format:

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

Example input: 5 Output: Automorphic Example input: 25 Output: Automorphic Example input: 7 Output: Not Automorphic

For example:

Test Result

print(automorphic(5)) Automorphic

Program:  
 **def check\_automorphic(n):**

**if n < 0:**

**return "Invalid input"**

**square = str(n \* n)**

**if square.endswith(str(n)):**

**return "Automorphic"**

**else:**

**return "Not Automorphic"**

**n = int(input("Enter a number: "))**

**print(check\_automorphic(n))**

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

**Register No.: 231501027 Name:K.Aswin**

**Check Product of Digits**

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

Input Format:

Take an input integer from stdin.

Output Format:

Print TRUE or FALSE.

Example Input:

1256

Output:

TRUE

Example Input:

1595

Output:

FALSE

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

For example:

**Program:**

**def check\_divisibility(n):**

**even\_product = 1**

**odd\_sum = 0**

**digits = [int(d) for d in str(n)]**

**for i, digit in enumerate(digits):**

**if i % 2 == 0:**

**even\_product \*= digit**

**else:**

**odd\_sum += digit**

**if odd\_sum == 0:**

**return False**

**else:**

**return even\_product % odd\_sum == 0**

**n = int(input("Enter a positive integer: "))**

**if check\_divisibility(n):**

**print("The product of digits at even places is divisible by the sum of digits at odd places.")**

**else:**

**print("The product of digits at even places is not divisible by the sum of digits at odd places.")**

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

**Register No.: 231501027 Name: K.Aswin**

**Christmas Discount**

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

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

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

**Constraints**

1 <= orderValue< 10e100000

Input

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

Output

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

Example Input

578

Output

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

12

**For example:**

**Program:**

**def is\_prime(n):**

**if n < 2:**

**return False**

**for i in range(2, int(n\*\*0.5) + 1):**

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

**return False**

**return True**

**def discount\_value(total\_bill):**

**discount = 0**

**for digit in str(total\_bill):**

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

**discount += int(digit)**

**return discount**

**total\_bill = int(input("Enter the total bill amount: "))**

**print("Discount value: ", discount\_value(total\_bill))**

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

**Register No.: 231501027 Name: K.Aswin**

**Coin Change**

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

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

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

Input Format:

Integer input from stdin.

Output Format:

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

Example Input:

16

Output:

4

Explanation:

We need only 4 coins of value 4 each

Example Input:

25

Output:

7

Explanation:

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

**Program:  
 def min\_coins(amount):**

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

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

**dp[0] = 0**

**for coin in coins:**

**for i in range(coin, amount + 1):**

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

**if dp[amount] == float('inf'):**

**return -1**

**else:**

**return dp[amount]**

**Ex. No. : 7.6 Date:**

**Register No.: 231501027 Name:K.Aswin**

**Difference Sum**

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

Input Format:

Take a number in the form of String from stdin.

Output Format:

Print the difference between sum of even and odd digits

Example input:

1453

Output:

1

Explanation:

Here, sum of even digits is 4 + 3 = 7

sum of odd digits is 1 + 5 = 6.

Difference is 1.

Note that we are always taking absolute difference

**Program:**

**def difference\_odd\_even\_position\_digits(n):**

**n = str(n)**

**sum\_odd = 0**

**sum\_even = 0**

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

**if i % 2 == 0:**

**sum\_even += int(n[i])**

**else:**

**sum\_odd += int(n[i])**

**return abs(sum\_odd - sum\_even)**

**:**

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

**Register No.: 231501027 Name:K.Aswin**

**Ugly number**

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

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

Task:

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

Hint:

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

Program: **For example:**

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

def is\_ugly(n):

if n <= 0:

return "not ugly"

while n % 2 == 0:

n = n // 2

while n % 3 == 0:

n = n // 3

while n % 5 == 0:

n = n // 5

if n == 1:

return "ugly"

else:

return "not ugly"

**08 – Tuple/Set**

Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

**For example:**

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

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

**Register No.: 231501027 Name:K.Aswin**

**Binary String**

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

**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 |

**Program:**

def count\_pairs(t, K):

seen = set()

for i in range(len(t)):

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

if t[i] + t[j] == K:

pair = tuple(sorted((t[i], t[j])))

seen.add(pair)

return len(seen)

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

**Register No.: 231501027 Name: K.Aswin**

**Check Pair**

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

**Example 1:**

**Input:** s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"

**Output:** ["AAAAACCCCC","CCCCCAAAAA"]

**Example 2:**

**Input:** s = "AAAAAAAAAAAAA"

**Output:** ["AAAAAAAAAA"]

| **Input** | **Result** |
| --- | --- |
| AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT | AAAAACCCCC  CCCCCAAAAA |

**For example:**

**Program:**

def count\_pairs(t, k):

count = 0

seen = set()

for i in range(len(t)):

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

if t[i] + t[j] == k:

pair = tuple(sorted((t[i], t[j])))

if pair not in seen:

count += 1

seen.add(pair)

return count

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

**Register No.: 231501027 Name: K.Aswin**

**DNA Sequence**

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

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

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

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

**Example 1:**

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

**Output:** 2

**Example 2:**

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

**Output:** 3

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

**For example:**

**Program:**

**def findRepeatedDnaSequences(s: str):**

**L, n = 10, len(s)**

**seen, output = {}, []**

**for i in range(n - L + 1):**

**tmp = s[i:i + L]**

**if tmp not in seen:**

**seen[tmp] = 1**

**else:**

**if seen[tmp] == 1:**

**output.append(tmp)**

**seen[tmp] += 1**

**return output**

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

**Register No.: 231501027 Name:K.Aswin**

**Print repeated no**

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.There is only **one repeated number** in nums, return this repeated number. Solve the problem using [set](http://118.185.187.137/moodle/mod/resource/view.php?id=734).

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

5 4

| **Input** | **Result** |
| --- | --- |
| 5 4  1 2 8 6 5  2 6 8 10 | 1 5 10  3 |

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

**Program:**

**def find\_repeated\_number(nums):**

**seen = set()**

**for num in nums:**

**if num in seen:**

**return num**

**seen.add(num)**

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

**print(find\_repeated\_number(nums))**

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

**print(find\_repeated\_number(nums))**

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

**Register No.: 231501027 Name: K.Aswin**

**Remove repeated**

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

Input Format:

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

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

Example 1:

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

Output:

1

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| hello world  ad | 1 |

Program:

def eliminate\_common\_elements(arr1, arr2):

set1 = set(arr1)

set2 = set(arr2)

non\_repeating\_elements = (set1 - set2) | (set2 - set1)

print(len(non\_repeating\_elements))

print(\*non\_repeating\_elements)

n, m = map(int, input().split())

arr1 = list(map(int, input().split()))

arr2 = list(map(int, input().split()))

eliminate\_common\_elements(arr1, arr2)

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

**Register No.: 231501027 Name: K.Aswin**

**Malfunctioning Keyboard**

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

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



**Example 1:**

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

**Output:** ["Alaska","Dad"]

**Example 2:**

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

**Output:** []

| **Input** | **Result** |
| --- | --- |
| 4  Hello  Alaska  Dad  Peace | Alaska  Dad |

**Example 3:**

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

**Output:** ["adsdf","sfd"]

**For example:**

Program:

def count\_words(text, broken\_letters):

broken\_set = set(broken\_letters)

count = 0

word = ""

for char in text:

if char == " ":

if not broken\_set.intersection(word):

count += 1

word = ""

else:

word += char

if not broken\_set.intersection(word):

count += 1

return count

text = "hello world"

broken\_letters = "ad"

print(count\_words(text, broken\_letters)) # Output: 1

text = "leet code"

broken\_letters = "lt"

print(count\_words(text, broken\_letters)) # Output: 1

text = "leet code"

broken\_letters = "e"

print(count\_words(text, broken\_letters))

**Ex. No. :8.7 Date:**

**Reg No: 231501027 Name: K.Aswin**

**American keyboard**

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

* 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 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".
* 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 |

**09 – Dictionary**

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

**Register No.: 231501027 Name: K.Aswin**

**Uncommon words**

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

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

**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 |

**Program:**

**def find\_uncommon\_words(s1, s2):**

**words1 = s1.split()**

**words2 = s2.split()**

**word\_count = {}**

**for word in words1 + words2:**

**if word in word\_count:**

**word\_count[word] += 1**

**else:**

**word\_count[word] = 1**

**uncommon\_words = [word for word, count in word\_count.items() if count == 1]**

**uncommon\_words.sort()**

**return uncommon\_words**

**def sum\_scores(test\_dict):**

**for key, values in test\_dict.items():**

**test\_dict[key] = sum(values)**

**return test\_dict**

**test\_dict = {'Gfg' : [6, 7, 4], 'best' : [7, 6, 5]}**

**print(sum\_scores(test\_dict)) # Output: {'Gfg': 17, 'best': 18}**

**test\_dict = {'Gfg' : [8,8], 'best' : [5,5]}**

**print(sum\_scores(test\_dict)) # Output: {'best': 10, 'Gfg': 16}**

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

**Register No.: 231501027 Name: K.Aswin**

**Sort Dictionary by Values Summation**

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

**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

| **Input** | **Result** |
| --- | --- |
| 10  John  John  Johny  Jamie  Jamie  Johny  Jack  Johny  Johny  Jackie | Johny |

**Sample Output:**

Johny

**For example:**

**Program:**

**def find\_winner(votes):**

**vote\_count = {}**

**for vote in votes:**

**if vote in vote\_count:**

**vote\_count[vote] += 1**

**else:**

**vote\_count[vote] = 1**

**sorted\_candidates = sorted(vote\_count.items(), key=lambda x: (-x[1], x[0]))**

**return sorted\_candidates[0][0]**

**votes = ["john", "johnny", "jackie", "johnny", "john", "jackie", "jamie", "jamie", "john", "johnny", "jamie", "johnny", "john"]**

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

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

**Register No.: 231501027 Name: K.Aswin**

**Winner of Election**

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

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

**Program:**

**def find\_winners(votes):**

**vote\_count = {}**

**for i, vote in enumerate(votes):**

**name, \*score = vote.split()**

**if name in vote\_count:**

**vote\_count[name] += sum(map(int, score))**

**else:**

**vote\_count[name] = sum(map(int, score))**

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

**winners = [candidate for candidate, count in vote\_count.items() if count == max\_votes]**

**winners.sort()**

**return winners**

**votes = ["James 67 89 56", "Lalith 89 45 45", "Ram 89 89 89", "Sita 70 70 70"]**

**winners = find\_winners(votes)**

**for winner in winners:**

**print(winner)**

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

**Register No.: 231501027 Name: K.Aswin**

**Student Record**

Create a student dictionary  for n students with the student name as key and their test mark assignment mark and lab mark as values. Do the following computations and display the result.

1.Identify the student with the  highest [average](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5717) 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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5717) score

Note:

If more than one student has the same score display all the student names

The points associated with each letter are shown below:

Points Letters

1 A, E, I, L, N, O, R, S, T and U

2 D and G

3 B, C, M and P

4 F, H, V, W and Y

5 K

8 J and X

10 Q and Z

[Sample](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.

Program:

def create\_student\_dict(n):

student\_dict = {}

for i in range(n):

name = input(f"Enter student {i+1} name: ")

test\_mark = float(input(f"Enter {name}'s test mark: "))

assignment\_mark = float(input(f"Enter {name}'s assignment mark: "))

lab\_mark = float(input(f"Enter {name}'s lab mark: "))

student\_dict[name] = {"test": test\_mark, "assignment": assignment\_mark, "lab": lab\_mark}

return student\_dict

def compute\_average\_marks(student\_dict):

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

average\_mark = (marks["test"] + marks["assignment"] + marks["lab"]) / 3

student\_dict[student]["average"] = average\_mark

return student\_dict

def identify\_top\_students(student\_dict, key):

top\_students = []

max\_mark = max(marks[key] for marks in student\_dict.values())

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

if marks[key] == max\_mark:

top\_students.append(student)

return top\_students

def identify\_bottom\_students(student\_dict, key):

bottom\_students = []

min\_mark = min(marks[key] for marks in student\_dict.values())

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

if marks[key] == min\_mark:

bottom\_students.append(student)

return bottom\_students

n = int(input("Enter the number of students: "))

student\_dict = create\_student\_dict(n)

student\_dict = compute\_average\_marks(student\_dict)

print("Student Dictionary:")

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

print(f"{student}: {marks}")

highest\_average\_students = identify\_top\_students(student\_dict, "average")

print(f"Students with the highest average score: {', '.join(highest\_average\_students)}")

highest\_assignment\_students = identify\_top\_students(student\_dict, "assignment")

print(f"Students with the highest assignment marks: {', '.join(highest\_assignment\_students)}")

lowest\_lab\_students = identify\_bottom\_students(student\_dict, "lab")

print(f"Students with the lowest lab marks: {', '.join(lowest\_lab\_students)}")

lowest\_average\_students = identify\_bottom\_students(student\_dict, "average")

print(f"Students with the lowest average score: {', '.join(lowest\_average\_students)}")

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

**Register No.: 231501027 Name: K.Aswin**

[**Scramble Score**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5780)

In the game of Scrabble™, each letter has points associated with it. The total score of a word is the sum of the scores of its letters. More common letters are worth fewer points while less common letters are worth more points.

Write a program that computes and displays the Scrabble™ score for a word. Create a dictionary that maps from letters to point values. Then use the dictionary to compute the score.

A Scrabble™ board includes some squares that multiply the value of a letter or the value of an entire word. We will ignore these squares in this exercise.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 5  6 5 4 3 8 | 3 4 5 6 8 |

Program:

scrabble\_points = {

'a': 1, 'b': 3, 'c': 3, 'd': 2, 'e': 1, 'f': 4,

'g': 2, 'h': 4, 'i': 1, 'j': 8, 'k': 5, 'l': 1,

'': 3, 'n': 1, 'o': 1, 'p': 3, 'q': 10, 'r': 1,

'': 1, 't': 1, 'u': 1, 'v': 4, 'w': 4, 'x': 8,

'y': 4, 'z': 10

}

def compute\_scrabble\_score(word):

score = 0

for letter in word.lower():

if letter in scrabble\_points:

score += scrabble\_points[letter]

return score

word = input("Enter a word: ")

score = compute\_scrabble\_score(word)

print(f"The Scrabble score for '{word}' is {score}.")

**10 . SEARCHING AND SORTING**

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

**Register No.: 231501027 Name: K.Aswin**

**Merge Sort**

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

**Input Format**

The first line contains an integer,n , the size of the [list](http://118.185.187.137/moodle/mod/resource/view.php?id=1068) a .  
The second line contains  n,  space-separated integers a[i].

**Constraints**

·         2<=n<=600

·         1<=a[i]<=2x106.

**Output Format**

You must print the following three lines of output:

1.      [List](http://118.185.187.137/moodle/mod/resource/view.php?id=1068) is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.

2.      First Element: firstElement, the  *first* element in the sorted [list](http://118.185.187.137/moodle/mod/resource/view.php?id=1068).

3.      Last Element: lastElement, the *last* element in the sorted [list](http://118.185.187.137/moodle/mod/resource/view.php?id=1068).

**Sample Input 0**

3 **For example:**

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

1 2 3

**Sample Output 0**

[List](http://118.185.187.137/moodle/mod/resource/view.php?id=1068) is sorted in 0 swaps.

First Element: 1

Last Element: 3

Program:

def merge\_sort(arr):

if len(arr) <= 1:

return arr, 0

mid = len(arr) // 2

left, num\_swaps\_left = merge\_sort(arr[:mid])

right, num\_swaps\_right = merge\_sort(arr[mid:])

merged, num\_swaps\_merged = merge(left, right)

num\_swaps = num\_swaps\_left + num\_swaps\_right + num\_swaps\_merged

return merged, num\_swaps

def merge(left, right):

merged = []

i = j = swaps = 0

while i < len(left) and j < len(right):

if left[i] <= right[j]:

merged.append(left[i])

i += 1

else:

merged.append(right[j])

j += 1

swaps += len(left) - i

merged.extend(left[i:])

merged.extend(right[j:])

return merged, swaps

# Test the function

arr = [6, 4, 1]

sorted\_arr, num\_swaps = merge\_sort(arr)

print(f"Array is sorted in {num\_swaps} swaps.")

print(f"First Element: {sorted\_arr[0]}")

print(f"Last Element: {sorted\_arr[-1]}")

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

**Register No.: 231501027 Name: K.Aswin**

**Bubble Sort**

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

1.      [List](http://118.185.187.137/moodle/mod/resource/view.php?id=1068) is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.

2.      First Element: firstElement, the  *first* element in the sorted [list](http://118.185.187.137/moodle/mod/resource/view.php?id=1068).

3.      Last Element: lastElement, the *last* element in the sorted [list](http://118.185.187.137/moodle/mod/resource/view.php?id=1068).

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

Array is sorted in 3 swaps.

First Element: 1

Last Element: 6

**Input Format**

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

**Output Format**

**Print** peak numbers separated by space.

**Sample Input**

5

8 9 10 2 6

**Sample Output**

10 6

**For example:**

| **Input** | **Result** |
| --- | --- |
| 4  12 3 6 8 | 12 8 |

Program:

def bubble\_sort(arr):

n = len(arr)

num\_swaps = 0

for i in range(n):

swapped = False

for j in range(n - 1):

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

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

num\_swaps += 1

swapped = True

if not swapped:

break

print(f"Array is sorted in {num\_swaps} swaps.")

print(f"First Element: {arr[0]}")

print(f"Last Element: {arr[-1]}")

# Test the function

arr = [6, 4, 1]

bubble\_sort(arr)

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

**Register No.: 231501027 Name: K.Aswin**

**Peak Element**

Given an [list](http://118.185.187.137/moodle/mod/resource/view.php?id=1068), 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]

**For example:**

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

Program:

def find\_peak\_element(arr):

n = len(arr)

if n == 1:

return 0

if arr[0] >= arr[1]:

return 0

if arr[n-1] >= arr[n-2]:

return n-1

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

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

return i

return -1

arr = [1, 3, 20, 4, 1, 0]

index = find\_peak\_element(arr)

if index != -1:

print("Peak element is", arr[index], "at index", index)

else:

print("No peak element found")

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

**Register No.: 2315010227 Name: K.Aswin**

**Binary Search**

Write a Python program for binary search.

Program:

def binary\_search(arr, low, high, x):

if high >= low:

mid = (high + low) // 2

if arr[mid] == x:

return mid

elif arr[mid] > x:

return binary\_search(arr, low, mid - 1, x)

else:

return binary\_search(arr, mid + 1, high, x)

else:

return -1

# Test the function

arr = [2, 3, 4, 10, 40]

x = 10

result = binary\_search(arr, 0, len(arr)-1, x)

if result != -1:

print("Element is present at index", str(result))

else:

print("Element is not present in array")

**Input:**

1 68 79 4 90 68 1 4 5

**output:**

 1 2

 4 2

 5 1

 68 2

 79 1

90 1

**For example:**

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

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

**Register No.: 231501027 Name: K.Aswin**

**Frequency of Elements**

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

**Constraints:**

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

Program:

from collections import Counter

n = int(input())

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

freq = Counter(arr)

sorted\_freq = dict(sorted(freq.items()))

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

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