RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE

CS23221
PYTHON PROGRAMMING LAB

Laboratory Observation Note Book

Name : JAGDISH KHANNAA K P

Year / Branch / Section : 1/AIML/B

Register No. : 231501064

Semester : II

Academic Year : 2023-2024

INDEX

Reg. No. : 231501064 Name : JAGDISH KHANNAA K P

Year : 1 Branch : AIML Sec : B

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.: **Name:**

Converting Input Strings

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

Sample Output:

10,<class 'int'>

10.9,<class 'float'>

For example:

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

```
a=input()
b=input()
c=int(a)
d=float(b)
print(c,type(c),sep=",")
print("{:0.1f}".format(d),type(d),sep=",")
```

	Input	Expected	Got	
✔	10 10.9	10,<class 'int'> 10.9,<class 'float'>	10,<class 'int'> 10.9,<class 'float'>	✔
✔	12 12.5	12,<class 'int'> 12.5,<class 'float'>	12,<class 'int'> 12.5,<class 'float'>	✔
✔	89 7.56	89,<class 'int'> 7.6,<class 'float'>	89,<class 'int'> 7.6,<class 'float'>	✔
✔	55000 56.2	55000,<class 'int'> 56.2,<class 'float'>	55000,<class 'int'> 56.2,<class 'float'>	✔
✔	2541 2541.679	2541,<class 'int'> 2541.7,<class 'float'>	2541,<class 'int'> 2541.7,<class 'float'>	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Gross Salary

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

Sample Input:

10000

Sample Output:

16000

For example:

Input	Result
10000	16000

```
s=int(input())
da=s*0.4
ha=s*0.2
print(int(s+da+ha))
```

	Input	Expected	Got	
✔	10000	16000	16000	✔
✔	20000	32000	32000	✔
✔	28000	44800	44800	✔
✔	5000	8000	8000	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Square Root

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

Sample Input:

8.00

Sample Output:

2.828

For example:

Input	Result
14.00	3.742

```
import math
a=float(input())
s=math.sqrt(a)
print("{:.3f}".format(s))
```

	Input	Expected	Got	
✔	8.00	2.828	2.828	✔
✔	14.00	3.742	3.742	✔
✔	4.00	2.000	2.000	✔
✔	487	22.068	22.068	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Gain percent

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

Input Format:

The first line contains the Rs X

The second line contains Rs Y

The third line contains Rs Z

Sample Input:

10000

250

15000

Sample Output:

46.34 is the gain percent.

For example:

Input	Result
45500 500 60000	30.43 is the gain percent.

```
buys=int(input())
repair=int(input())
sells=int(input())
g=(((sells-(buys+repair))/(buys+repair))*100)
print("{:.2f}".format(g), "is the gain percent.")
```

	Input	Expected	Got	
✔	10000 250 15000	46.34 is the gain percent.	46.34 is the gain percent.	✔
✔	45500 500 60000	30.43 is the gain percent.	30.43 is the gain percent.	✔
✔	5000 0 7000	40.00 is the gain percent.	40.00 is the gain percent.	✔
✔	12500 5000 18000	2.86 is the gain percent.	2.86 is the gain percent.	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Deposits

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

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.

```
a=int(input())
b=int(input())
c=a*0.1
d=b*0.25
e=c+d
print("Your total refund will be ${:.2f}.".format(e))
```

	Input	Expected	Got	
✔	20 20	Your total refund will be $7.00.	Your total refund will be $7.00.	✔
✔	11 22	Your total refund will be $6.60.	Your total refund will be $6.60.	✔
✔	123 200	Your total refund will be $62.30.	Your total refund will be $62.30.	✔
✔	76 38	Your total refund will be $17.10.	Your total refund will be $17.10.	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Carpenter

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

Hint:

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

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

```
number = -20
absolute_number = abs(number)
```

```
print(absolute_number)
# Output:20
```

Sample Input:

450

Sample Output:

weekdays 10.38

weekend 0.38

For example:

Input	Result
450	weekdays 10.38 weekend 0.38

```
s=int(input())
a=(500-s)/130
print("weekdays {:.2f}".format(abs(a)+10))
print("weekend {:.2f}".format(abs(a)))
```

	Input	Expected	Got	
✔	450	weekdays 10.38 weekend 0.38	weekdays 10.38 weekend 0.38	✔
✔	500	weekdays 10.00 weekend 0.00	weekdays 10.00 weekend 0.00	✔
✔	10000	weekdays 83.08 weekend 73.08	weekdays 83.08 weekend 73.08	✔
✔	6789	weekdays 58.38 weekend 48.38	weekdays 58.38 weekend 48.38	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

02- Operators in Python

Ex. No. : 2.1 Date:

Register No.: **Name:**

Widgets and Gizmos

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

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.

```
a=int(input())

b=int(input())

print("The total weight of all these widgets and gizmos is",((a*75)+(b*112)),"grams.")
```

	Input	Expected	Got
✔	10 20	The total weight of all these widgets and gizmos is 2990 grams.	The to

Passed all tests! ✔

Correct

Marks for this submission: 10.00/10.00.

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

Register No.: **Name:**

Doll Sings

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

IF Lokpaul wins print true, otherwise false.

Sample Input

10

Sample Output

True

Explanation:

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

```
a=int(input())
```

```
if(a>0 and a<100 and a%2==0):
    print("True")
else:
    print("False")
```

	Input	Expected	Got	
✔	56	True	True	✔
✔	101	False	False	✔
✔	-1	False	False	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Birthday Party

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

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

```
a=int(input())
b=int(input())
c=int(input())
d=int(input())
e=int(input())
print(b%a==0,c%a==0,d%a==0,e%a==0)
```

	Input	Expected	Got	
✔	5 25 23 20 10	True False True True	True False True True	✔
✔	4 23 24 21 12	False True False True	False True False True	✔
✔	8 64 8 16 32	True True True True	True True True True	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Ex. No. : 2.4 Date:

Register No.: **Name:**

Hamming Weight

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

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.

```
a=int(input())

n=bin(a)

n=n.replace("0b","")

s=str(n)

c=list(s)

d=0

for i in range(len(c)):

   if(int(c[i])==1):

       d+=1

print(d)
```

	Input	Expected	Got	
✔	3	2	2	✔
✔	5	2	2	✔
✔	15	4	4	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Compound Interest

Pretend that you have just opened a new savings account that earns 4 percent interest per year. The interest that you earn is paid at the end of the year, and is added

to the balance of the savings account. Write a program that begins by reading the amount of money deposited into the account from the user. Then your program should compute and display the amount in the savings account after 1, 2, and 3 years. Display each amount so that it is rounded to 2 decimal places.

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

```
a=int(input())

b=(a*0.04)+a

c=b+(b*0.04)

d=c+(c*0.04)

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

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

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

	Input	Expected	Got
✔	10000	Balance as of end of Year 1: $10400.00. Balance as of end of Year 2: $10816.00. Balance as of end of Year 3: $11248.64.	Balance as of end of Year 1: $ Balance as of end of Year 2: $ Balance as of end of Year 3: $
✔	20000	Balance as of end of Year 1: $20800.00. Balance as of end of Year 2: $21632.00. Balance as of end of Year 3: $22497.28.	Balance as of end of Year 1: $ Balance as of end of Year 2: $ Balance as of end of Year 3: $

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Eligible to donate blood

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

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

Input Format:

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

Output Format:

Display True(IF ELIGIBLE)

Display False (if not eligible)

Sample Input

19

45

Sample Output

True

```
a=int(input())
b=int(input())
if(a>=18 and b>40):
    print("True")
else:
```

```
print("False")
```

	Input	Expected	Got	
✔	19 45	True	True	✔
✔	18 40	False	False	✔
✔	18 42	True	True	✔
✔	16 45	False	False	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

C or D

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

Hint:

Use ASCII values of C and D.

Input Format:

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

Output Format:

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

Input 1:
0
Output 1:
C

Input 2:
1

Output 1:
D

```
a=int(input())
if(a==0):
    print("C")
else:
    print("D")
```

	Input	Expected	Got	
✔	0	C	C	✔
✔	1	D	D	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Troy Battle

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

Input format:

Line 1 has the total number of weapons

Line 2 has the total number of Soldiers.

Output Format:

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

Sample Input:

32

43

Sample Output:'

False

```
a=int(input())
b=int(input())
if(a%3==0 and b%2==0):
    print("True")
else:
    print("False")
```

	Input	Expected	Got	
✔	32 43	False	False	✔
✔	273 7890	True	True	✔
✔	800 4590	False	False	✔
✔	6789 32996	True	True	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Tax and Tip

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

Sample Input

100

Sample Output

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

```
a=int(input())
print("The tax is {:.2f} and the tip is {:.2f}, making the total {:.2f}".format((a*0.05),(a*0.18),(a+((a*0.05)+(a*0.18)))))
```

	Input	Expected	Got
✔	100	The tax is 5.00 and the tip is 18.00, making the total 123.00	The tax
✔	250	The tax is 12.50 and the tip is 45.00, making the total 307.50	The tax

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Return last digit of the given number

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

The last digit should be returned as a positive number.

For example,

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

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

For example:

Input	Result
123	3

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

	Input	Expected	Got	
✔	197	7	7	✔
✔	-197	7	7	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

03 - Selection Structures in Python

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

Register No.: **Name:**

Admission Eligibility

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

Marks in Maths >= 65

Marks in Physics >= 55

Marks in Chemistry >= 50

Or

Total in all three subjects >= 180

Sample Test Cases

Test Case 1

Input

70

60

80

Output

The candidate is eligible

Test Case 2

Input

50

80

80

Output

The candidate is eligible

Test Case 3

Input

50

60

40

Output

The candidate is not eligible

For example:

Input	Result
50 80 80	The candidate is eligible

```
a=int(input())
b=int(input())
c=int(input())
if(a>=65 and b>=55 and c>=50):
    print("The candidate is eligible")
elif(a+b+c>=180):
    print("The candidate is eligible")
else:
    print("The candidate is not eligible")
```

	Input	Expected	Got	
✔	70 60 80	The candidate is eligible	The candidate is eligible	✔
✔	50 80 80	The candidate is eligible	The candidate is eligible	✔
✔	50 60 40	The candidate is not eligible	The candidate is not eligible	✔
✔	20 10 25	The candidate is not eligible	The candidate is not eligible	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Classifying Triangles

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

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

Sample Input 1

60

60

60

Sample Output 1

That's a equilateral triangle

For example:

Input	Result
40 40 80	That's a isosceles triangle

```
a=int(input())
b=int(input())
c=int(input())
if(a==b and b==c):
    print("That's a equilateral triangle")
elif(a!=b and b==c or a==b and b!=c):
    print("That's a isosceles triangle")
elif(a!=b and b!=c):
    print("That's a scalene triangle")
```

	Input	Expected	Got	
✔	60 60 60	That's a equilateral triangle	That's a equilateral triangle	✔
✔	40 40 80	That's a isosceles triangle	That's a isosceles triangle	✔
✔	50 60 70	That's a scalene triangle	That's a scalene triangle	✔
✔	50 50 80	That's a isosceles triangle	That's a isosceles triangle	✔
✔	10 10 10	That's a equilateral triangle	That's a equilateral triangle	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Electricity Bill

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

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

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

Sample Test Cases

Test Case 1

Input

50

Output

100.00

Test Case 2

Input

300

Output

517.50

For example:

Input	Result
500	1035.00

```
a=float(input())
b=0
if(a<=199):
    b=a*1.2
elif(200<=a<400):
    b=a*1.5
elif(400<=a<600):
    b=a*1.8
elif(a>600):
    b=a*2.0
```

```
if (int(b)<100):
    print("{:.2f}".format(100))
else:
    if(b>400.00):
        print("{:.2f}".format((b+(b*0.15))))
    else:
        print("{:.2f}".format(b))
```

	Input	Expected	Got	
✔	50	100.00	100.00	✔
✔	100.00	120.00	120.00	✔
✔	500	1035.00	1035.00	✔
✔	700	1610.00	1610.00	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

IN/OUT

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

Input Format:

Input consists of 2 integers.

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

Output Format:

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

Sample Input and Output:

Input

8

3

Output

OUT

For example:

Input	Result
8 3	OUT

```
a=int(input())
b=int(input())
c=(a/2)
if(c>b):
    print("OUT")
else:
    print("IN")
```

	Input	Expected	Got	
✔	8 3	OUT	OUT	✔
✔	8 5	IN	IN	✔
✔	20 9	OUT	OUT	✔
✔	50 31	IN	IN	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Vowel or Consonant

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

Sample Input 1

i

Sample Output 1

It's a vowel.

Sample Input 2

y

Sample Output 2

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

Sample Input3

c

Sample Output 3

It's a consonant.

For example:

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

```
a=input()
if(a=='a' or a=='e' or a=='i' or a=='o' or a=='u'):
    print("It's a vowel.")
elif(a=='y'):
    print("Sometimes it's a vowel... Sometimes it's a consonant.")
else:
    print("It's a consonant.")
```

	Input	Expected	Got
✔	i	It's a vowel.	It's a vowel.
✔	y	Sometimes it's a vowel... Sometimes it's a consonant.	Sometimes it's a
✔	c	It's a consonant.	It's a consonant
✔	e	It's a vowel.	It's a vowel.
✔	r	It's a consonant.	It's a consonant

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Leap Year

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

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

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

Sample Input 1

1900

Sample Output 1

1900 is not a leap year.

Sample Input 2

2000

Sample Output 2

2000 is a leap year.

```
year=int(input())
if(year%400==0):
    print(year,"is a leap year.")
else:
    print(year,"is not a leap year.")
```

	Input	Expected	Got	
✔	1900	1900 is not a leap year.	1900 is not a leap year.	✔
✔	2000	2000 is a leap year.	2000 is a leap year.	✔
✔	2100	2100 is not a leap year.	2100 is not a leap year.	✔
✔	2400	2400 is a leap year.	2400 is a leap year.	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Month name to days

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

Sample Input 1

February

Sample Output 1

February has 28 or 29 days in it.

Sample Input 2

March

Sample Output 2

March has 31 days in it.

Sample Input 3

April

Sample Output 3

April has 30 days in it.

For example:

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

Input	Result
March	March has 31 days in it.

```
m=input()
if(m=="January"):
    print(m,"has 31 days in it.")
elif(m=="February"):
    print(m,"has 28 or 29 days in it.")
elif(m=="March"):
    print(m,"has 31 days in it.")
elif(m=="April"):
    print(m,"has 30 days in it.")
elif(m=="May"):
    print(m,"has 31 days in it.")
elif(m=="June"):
    print(m,"has 30 days in it.")
elif(m=="July"):
    print(m,"has 31 days in it.")
elif(m=="August"):
    print(m,"has 31 days in it.")
elif(m=="September"):
    print(m,"has 30 days in it.")
```

```
elif(m=="October"):
    print(m,"has 31 days in it.")
elif(m=="November"):
    print(m,"has 30 days in it.")
elif(m=="December"):
    print(m,"has 31 days in it.")
```

	Input	Expected	Got
✔	February	February has 28 or 29 days in it.	February has 28 or 29 days in it.
✔	March	March has 31 days in it.	March has 31 days in it.
✔	April	April has 30 days in it.	April has 30 days in it.
✔	May	May has 31 days in it.	May has 31 days in it.

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Pythagorean triple

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

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

Sample Input

3

5

4

Sample Output

Yes

For example:

Input	Result
3 4 5	Yes

```
a=int(input())

b=int(input())

c=int(input())

if(a*a+b*b==c*c):

    print("yes")
```

```
elif(a*a+c*c==b*b):
    print("yes")
elif(c*c+b*b==a*a):
    print("yes")
else:
    print("no")
```

	Input	Expected	Got	
✔	3 5 4	yes	yes	✔
✔	5 8 2	no	no	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Second last digit

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

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

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

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

For example:

Input	Result
197	9

```
a=int(input())
b=str(abs(a))
l=len(b)
if(l>1):
    print(int(b[-2]))
else:
    print(-1)
```

	Input	Expected	Got	
✔	197	9	9	✔
✔	-197	9	9	✔
✔	5	-1	-1	✔
✔	123456	5	5	✔
✔	8	-1	-1	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Chinese Zodiac

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

Year Animal

2000 Dragon

2001 Snake

2002 Horse

2003 Sheep

2004 Monkey

2005 Rooster

2006 Dog

2007 Pig

2008 Rat

2009 Ox

2010 Tiger

2011 Hare

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

Sample Input 1

2010

Sample Output 1

2010 is the year of the Tiger.

Sample Input 2

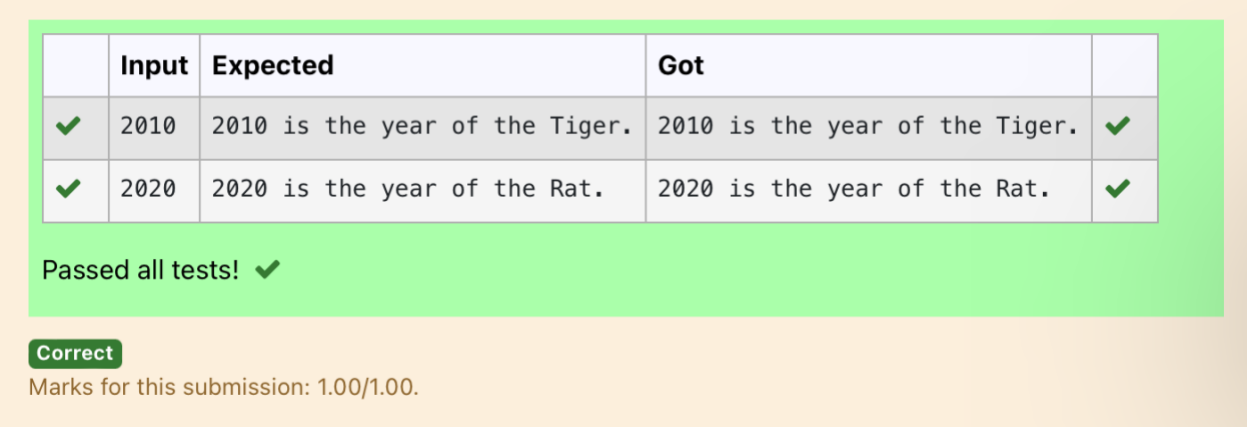
2020

Sample Output 2

2020 is the year of the Rat.

```
a=int(input())
b=a%100
c=b%12
if(c==0):
    print(a,"is the year of the Dragon.")
elif(c==1):
    print(a,"is the year of the Snake.")
elif(c==2):
    print(a,"is the year of the Horse.")
elif(c==3):
    print(a,"is the year of the Sheep.")
elif(c==4):
    print(a,"is the year of the Monkey.")
elif(c==5):
    print(a,"is the year of the Rooster.")
elif(c==6):
    print(a,"is the year of the Dog.")
elif(c==7):
    print(a,"is the year of the Pig.")
elif(c==8):
    print(a,"is the year of the Rat.")
```

```
elif(c==9):
    print(a,"is the year of the Ox.")
elif(c==10):
    print(a,"is the year of the Tiger.")
elif(c==11):
    print(a,"is the year of the Hare.")
```



	Input	Expected	Got	
✔	2010	2010 is the year of the Tiger.	2010 is the year of the Tiger.	✔
✔	2020	2020 is the year of the Rat.	2020 is the year of the Rat.	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

04 - Iteration Control Structures

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

Register No.: **Name:**

Factors of a number

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

For example:

Input	Result
20	1 2 4 5 10 20

```
k=int(input())

l=[]

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

    if(k%i==0):

        l.append(i)

for j in l:

    print(j,end=' ')
```

	Input	Expected	Got	
✔	20	1 2 4 5 10 20	1 2 4 5 10 20	✔
✔	5	1 5	1 5	✔
✔	13	1 13	1 13	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Non Repeated Digit Count

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

For example:

Input	Result
292	1
1015	2
108	3
22	0

```
n=int(input())
l=[]
k=[]
while n>0:
    a=n%10
    n=n//10
    l.append(a)
for i in range(len(l)):
    if l.count(l[i])==1:
        k.append(l[i])
```

```
print(len(k))
```

	Input	Expected	Got	
✔	292	1	1	✔
✔	1015	2	2	✔
✔	108	3	3	✔
✔	22	0	0	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Prime Checking

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

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

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

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

For example:

Input	Result
7	2
10	1

```
a=int(input())
for i in range(2,a):
    if(a%2==0):
        flag=0
    elif(a%i!=0):
        flag=1
    else:
        flag=0
if(flag==1):
    print("2")
elif(flag==0):
    print("1")
```

	Input	Expected	Got	
✔	7	2	2	✔
✔	10	1	1	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Next Perfect Square

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

Input Format:

Integer input from stdin.

Output Format:

Perfect square greater than N.

Example Input:

10

Output:

16

```
a=int(input())
c=[]
for i in range(0,a):
    b=i**2
    if(b>a):
        c.append(b)
print(c[0])
```

	Input	Expected	Got	
✔	10	16	16	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Nth Fibonacci

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

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

```
a=[0,1]
for i in range(0,100):
    a.append(a[-1]+a[-2])
```

```
q=int(input())
print(a[q-1])
```

	Input	Expected	Got	
✔	1	0	0	✔
✔	4	2	2	✔
✔	7	8	8	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Disarium Number

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

Input Format:

Single Integer Input from stdin.

Output Format:

Yes or No.

Example Input:

175

Output:

Yes

Explanation

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

Example Input:

123

Output:

No

For example:

Input	Result
175	Yes
123	No

import math

```
n=int(input())
a=len(str(n))
sum=0
x=n
while(x!=0):
    r=x%10
    sum=int(sum+math.pow(r,a))
    a-=1
    x=x//10
if(sum==n):
    print("Yes")
else:
    print("No")
```

	Input	Expected	Got	
✔	175	Yes	Yes	✔
✔	123	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Sum of Series

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

Sample Test Cases

Test Case 1

Input

4

Output

1234

Explanation:

as input is 4, have to take 4 terms.

1 + 11 + 111 + 1111

Test Case 2

Input

6

Output

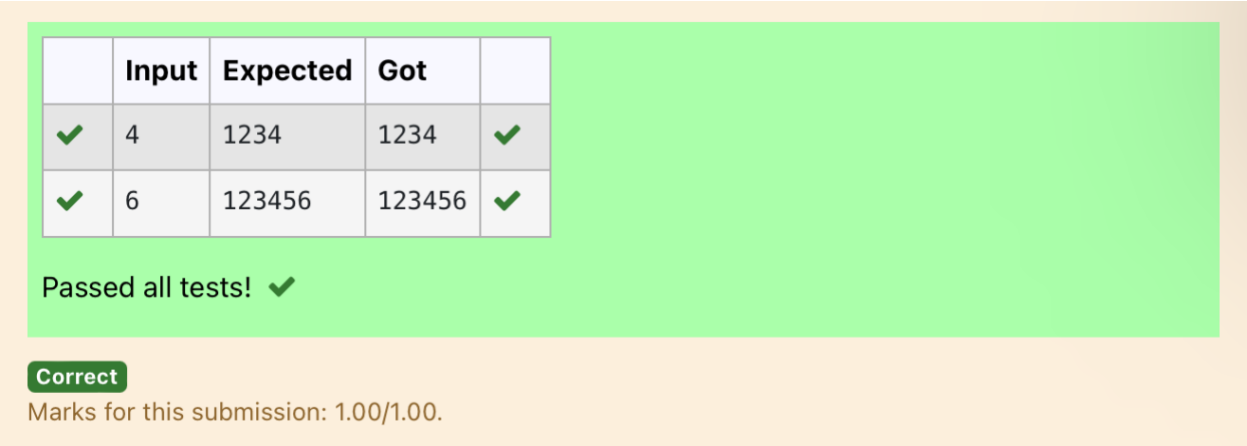
123456

For example:

Input	Result
3	123

```
n=int(input())

b=1

sum=0
```

```
for i in range(1,n+1):
    sum+=b
    b=(b*10)+1
print(sum)
```



	Input	Expected	Got	
✔	4	1234	1234	✔
✔	6	123456	123456	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Unique Digit Count

Write a program to find the count of unique digits in a given number N. The number will be passed to the program as an input of type int.
Assumption: The input number will be a positive integer number >= 1 and <= 25000.
For e.g.
If the given number is 292, the program should return 2 because there are only 2 unique digits '2' and '9' in this number
If the given number is 1015, the program should return 3 because there are 3 unique digits in this number, '1', '0', and '5'.
For example:

Input	Result
292	2
1015	3

```
a=int(input())
b=[]
while a>0:
    c=a%10
    a=a//10
    b.append(c)
b=list(set(b))
print(len(b))
```

	Input	Expected	Got	
✔	292	2	2	✔
✔	1015	3	3	✔
✔	123	3	3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Product of single digit

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

Input Format:

Single Integer input.

Output Format:

Output displays Yes if condition satisfies else prints No.

Example Input:

14

Output:

Yes

Example Input:

13

Output:

No

```
a=int(input())
flag=0
for i in range(10):
    for j in range(10):
        if(i*j==a):
            flag=1
            break
if(flag==1):
    print("Yes")
else:
    print("No")
```

	Input	Expected	Got	
✔	14	Yes	Yes	✔
✔	13	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Perfect Square After adding One

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

Input Format:

Single integer input.

Output Format:

Yes or No.

Example Input:

24

Output:

Yes

Example Input:

26

Output:

No

For example:

Input	Result
24	Yes

```
import math

n=int(input())

a=n+1
```

```
sr=int(math.sqrt(a))
if(sr*sr==a):
    print("Yes")
else:
    print("No")
```

	Input	Expected	Got	
✔	24	Yes	Yes	✔
✔	26	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

05 - List in Python

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

Register No.: **Name:**

Balanced Array

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

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

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

Constraints

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

The first line contains an integer n, the size of the array arr.
Each of the next n lines contains an integer, arr[i], where $0 \leq i < n$.
Sample Case 0
Sample Input 0
4
1
2
3
3
Sample Output 0
2
Explanation 0
The sum of the first two elements, 1+2=3. The value of the last element is 3
Using zero based indexing, arr[2]=3 is the pivot between the two subarrays
The index of the pivot is 2
Sample Case 1
Sample Input 1
3
1
2
1
Sample Output 1
1
Explanation 1
The first and last elements are equal to 1
Using zero based indexing, arr[1]=2 is the pivot between the two subarrays

The index of the pivot is 1.

For example:

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

```
a=int(input())
l=[]
for i in range(a):
    c=int(input())
    l.append(c)
for i in range(1,a):
    d=sum(l[0:i])
    r=sum(l[i+1:])
    if(d==r):
        print(i)
```

	Input	Expected	Got	
✔	4 1 2 3 3	2	2	✔
✔	3 1 2 1	1	1	✔

Passed all tests! ✔

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

Register No.: **Name:**

Check pair with difference k

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

Input Format

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

Output format

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

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

For example:

Input	Result
1 3 1 3 5 4	1

Input	Result
1 3 1 3 5 99	0

```
a=int(input())
while(a!=0):
    b=int(input())
    l=[]
    f=0
    for i in range(b):
        c=int(input())
        l.append(c)
    k=int(input())
    a-=1
    for i in range(b):
        for j in range(b):
            if(l[i]-l[j]==k and i!=j):
                f=1
                break
    if(f==1):
        print(1)
    else:
        print(0)
```

	Input	Expected	Got	
✔	1 3 1 3 5 4	1	1	✔
✔	1 3 1 3 5 99	0	0	✔

Passed all tests! ✔

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

Register No.: **Name:**

Count Elements

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

Sample Test Cases

Test Case 1

Input

7

23

45

23

56

45

23

40

Output

23 occurs 3 times

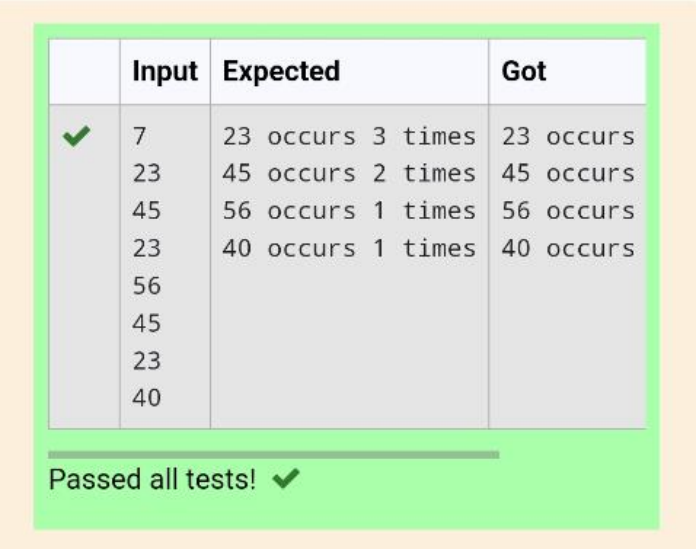
45 occurs 2 times

56 occurs 1 times

40 occurs 1 times

```
import collections

def CountFrequency(arr):

    return collections.Counter(arr)


if __name__ == "__main__":
```

```
# Input size of array
n = int(input())

# Input elements in array
arr = []
for _ in range(n):
ele = int(input())
arr.append(ele)

# Calculate frequency of each element
freq = CountFrequency(arr)


for key, value in freq.items():
print(f"{key} occurs {value} times")
```



	Input	Expected	Got
✔	7 23 45 23 56 45 23 40	23 occurs 3 times 45 occurs 2 times 56 occurs 1 times 40 occurs 1 times	23 occurs 45 occurs 56 occurs 40 occurs

Passed all tests! ✔

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

Register No.: **Name:**

Distinct Elements in an Array

Program to print all the distinct elements in an array. Distinct elements are nothing but the unique (non-duplicate) elements present in the given array.
Input Format:
First line take an Integer input from stdin which is array length n.
Second line take n Integers which is inputs of array.
Output Format:
Print the Distinct Elements in Array in single line which is space Separated

Example Input:
5
1
2
2
3
4
Output:
1 2 3 4
Example Input:
6
1
1
2
2
3
3
Output:
1 2 3

For example:
Input Result
5
1
2
2
3
4
1 2 3 4
6
1

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

```
def merge_arrays_without_duplicates(arr1, arr2):
    # Combine the arrays and convert to a set to remove duplicates
    result_set = set(arr1 + arr2)
    # Convert the set back to a sorted list
    merged_sorted_array = sorted(result_set)
    return merged_sorted_array

# Input read and processing
def process_input():
    # Reading number of elements and the elements for the first array
    n1 = int(input())
    array1 = []
    for _ in range(n1):
        element = int(input())
        array1.append(element)

    # Reading number of elements and the elements for the second array
    n2 = int(input())
    array2 = []
    for _ in range(n2):
```

```
    element = int(input())
    array2.append(element)

# Merge the arrays without duplicates
result = merge_arrays_without_duplicates(array1, array2)

# Print the result
print(" ".join(map(str, result)))
```

	Input	Expected
✔	5 1 2 3 6 9 4 2 4 5 10	1 2 3 4 5 6 9 10
✔	7 4 7 8 10 12 30 35 9 1 3 4 5 7 8 11 13 22	1 3 4 5 7 8 10 11 12 13 22 3(

Passed all tests! ✔

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

Register No.: **Name:**

Element Insertion

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

Sample Test Cases

Test Case 1

Input

1
3
4
5
6
7
8
9
10
11
2

Output

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

Test Case 2

Input

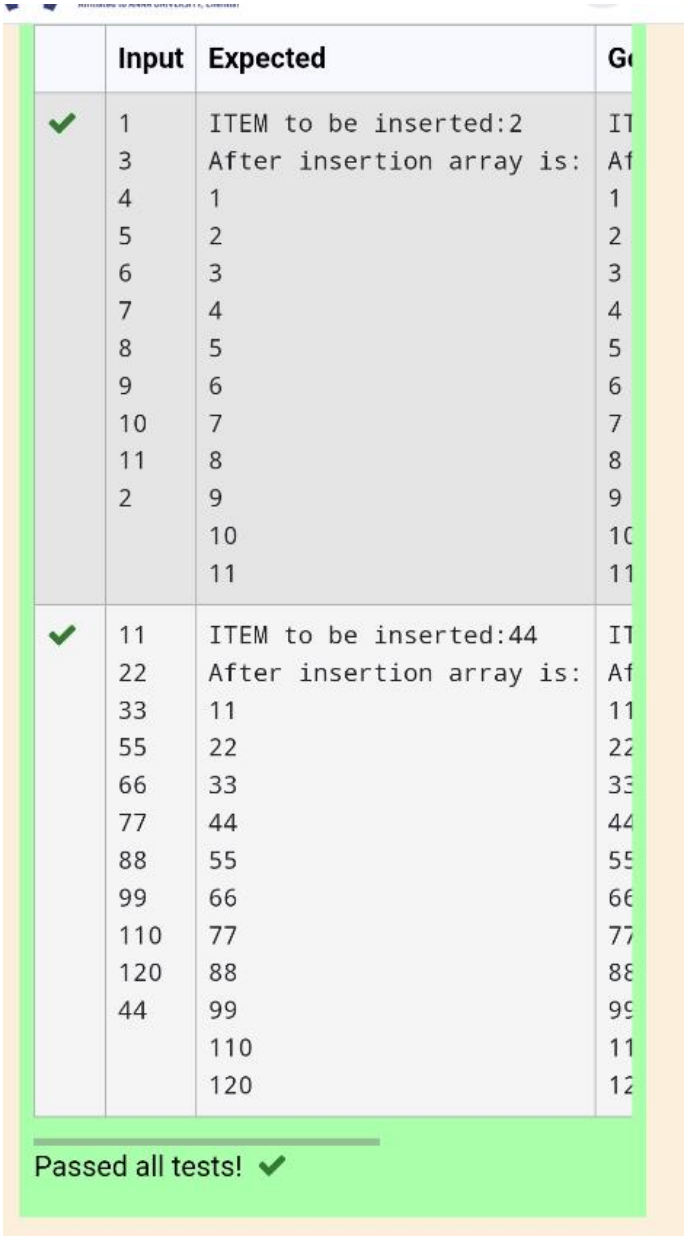
11
22
33
55
66
77
88
99
110
120
44

Output

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

```
def insert_sorted(list, n):
    list.append(n)
    sorted_list = sorted(list)
```

```
    print("After insertion array is:")
    for i in range(11):
        print(sorted_list[i])


sorted_list = [int(input()) for i in range(10)]
new_element = int(input())
print("ITEM to be inserted:", new_element, sep='')
insert_sorted(sorted_list, new_element)
```



	Input	Expected	G
✔	1 3 4 5 6 7 8 9 10 11 2	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	IT Af 1 2 3 4 5 6 7 8 9 10 11
✔	11 22 33 55 66 77 88 99 110 120 44	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	IT Af 11 22 33 44 55 66 77 88 99 11 12

Passed all tests! ✔

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

Register No.: **Name:**

Find the Factor

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

Constraints

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

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

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

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

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

For example:

Input	Result
10 3	5
10 5	0
1 1	1

```
import sys
import math

def find_factors(n):
    factors = []
    for i in range(1, int(math.sqrt(n)) + 1):
    if n % i == 0:
    factors.append(i)
    if i != n // i:
        factors.append(n // i)
    return sorted(factors)

def get_pth_factor(n, p):
    factors = find_factors(n)
    if p <= len(factors):
    return factors[p - 1]
    else:
```

```
    return 0


# Reading input directly from the standard input (typically for competitive programming)

input = sys.stdin.read

data = input().split()

n = int(data[0])

p = int(data[1])


# Calculate and print the p-th factor

print(get_pth_factor(n, p))
```

	Input	Expected	Got	
✔	10 3	5	5	✔
✔	10 5	0	0	✔
✔	1 1	1	1	✔

Passed all tests! ✔

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

Register No.: **Name:**

Merge List

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

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

Sample test case

Sample input
2
2
1
3
5
7
2
4
6
8

Sample Output

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

```
def zip_lists(list1, list2):
    return [row1 + row2 for row1, row2 in zip(list1, list2)]


def main():
    m = int(input())
```

```
    n = int(input())


    list1 = [[int(input()) for _ in range(n)] for _ in range(m)]
    list2 = [[int(input()) for _ in range(n)] for _ in range(m)]

    zipped_list = zip_lists(list1, list2)
    print(zipped_list)

if __name__ == "__main__":
    main()
```

	Input	Expected
✔	2 2 1 2 3 4 5 6 7 8	[[1, 2, 5, 6], [3, 4, 7, 8]]

Passed all tests! ✔

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

Register No.: **Name:**

Merge Two Sorted Arrays Without Duplication

Output is a merged array without duplicates.

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

Sample Input 1

5
1
2
3
6
9
4
2
4
5
10

Sample Output 1

1 2 3 4 5 6 9 10

```
def merge_arrays_without_duplicates(arr1, arr2):
    # Combine the arrays and convert to a set to remove duplicates
    result_set = set(arr1 + arr2)
    # Convert the set back to a sorted list
```

```
    merged_sorted_array = sorted(result_set)
    return merged_sorted_array

# Input read and processing
def process_input():
    # Reading number of elements and the elements for the first array
    n1 = int(input())
    array1 = []
    for _ in range(n1):
        element = int(input())
        array1.append(element)

    # Reading number of elements and the elements for the second array
    n2 = int(input())
    array2 = []
    for _ in range(n2):
        element = int(input())
        array2.append(element)

    # Merge the arrays without duplicates
    result = merge_arrays_without_duplicates(array1, array2)

    # Print the result
    print(" ".join(map(str, result)))
```

	Input	Expected
✔	5 1 2 3 6 9 4 2 4 5 10	1 2 3 4 5 6 9 10
✔	7 4 7 8 10 12 30 35 9 1 3 4 5 7 8 11 13 22	1 3 4 5 7 8 10 11 12 13 22 3(

Passed all tests! ✔

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

Register No.: **Name:**

Print Element Location

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

For example, if there are 4 elements in the array:
5
6
5
7

If the element to search is 5 then the output will be:
5 is present at location 1
5 is present at location 3
5 is present 2 times in the array.
Sample Test Cases

Test Case 1
Input

4
5
6
5
7
5

Output

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

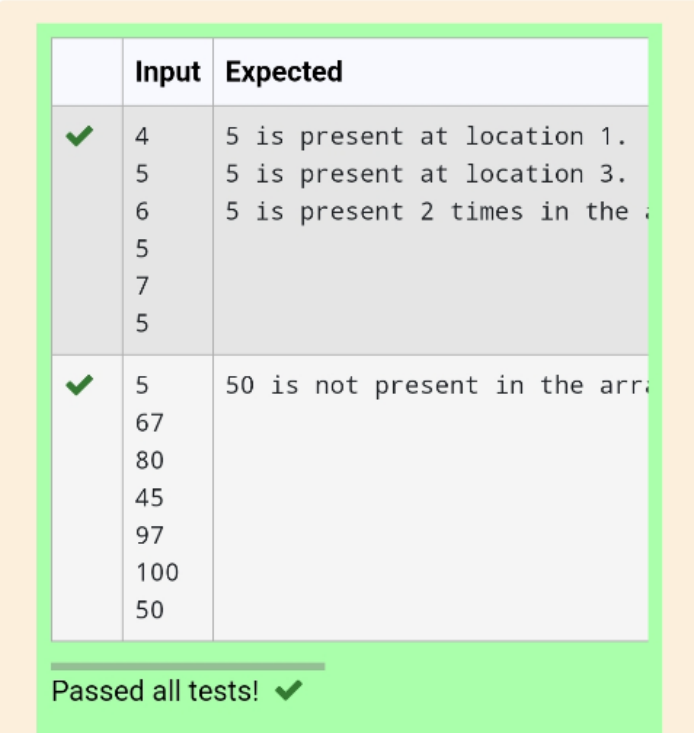
Test Case 2
Input
5
67
80

45
97
100
50

Output
50 is not present in the array.

```
def find_element_locations(lst, target):
    locations = []
    count = 0
    for i in range(len(lst)):
        if lst[i] == target:
            locations.append(i + 1)
            count += 1

    return locations, count

def main():
    n = int(input())
    lst = [int(input()) for _ in range(n)]
    target = int(input())

    locations, count = find_element_locations(lst, target)

    if count == 0:
        print(f"{target} is not present in the array.")
```

```
    else:
        for loc in locations:
            print(f"{target} is present at location {loc}.")
        print(f"{target} is present {count} times in the array.")

if __name__ == "__main__":
    main()
```



	Input	Expected
✔	4 5 6 5 7 5	5 is present at location 1. 5 is present at location 3. 5 is present 2 times in the a
✔	5 67 80 45 97 100 50	50 is not present in the arra

Passed all tests! ✔

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

Register No.: **Name:**

Strictly increasing

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

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

```
n= int(input())
arr = [int(input()) for i in range(n)]
```

```
l = arr.copy()
g=0
size = len(arr)
arr_asc = sorted(arr)
arr_des = sorted(arr)[::-1]
if arr==arr_asc or arr==arr_des:
    print('True')
    g=1
else:
    for i in arr:
        l.remove(i)
        arr_asc.remove(i)
        arr_des.remove(i)
        if l==arr_asc or l==arr_des:
            print('True')
            g=1
            break
        l=arr.copy()
        arr_asc = sorted(arr)
        arr_des = sorted(arr)[::-1]
if g==0:
    print('False')
```

	Input	Expected	Got	
✔	7 1 2 3 0 4 5 6	True	True	✔
✔	4 2 1 0 -1	True	True	✔

Passed all tests! ✔

06 - Strings in Python

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

Register No.: **Name:**

Count Chars

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

For example:

Input Result
rec@123
3
3
1

```
a=input()
c,d,s=0,0,0
for i in range(len(a)):
    if(a[i].isalpha()):
        c+=1
    elif(a[i].isdigit()):
        d+=1
    else:
        s+=1
print(c,d,s,sep="\n")
```

	Input	Expected	Got	
✔	rec@123	3 3 1	3 3 1	✔
✔	P@#yn26at^&i5ve	8 3 4	8 3 4	✔
✔	abc@12&	3 2 2	3 2 2	✔

Passed all tests! ✔

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

Register No.: **Name:**

Decompress the String

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

Sample Input 1
a2b4c6

Sample Output 1
aabbbbccccccc

```
import re
a=input()
all=re.findall('\d+',a)
all_w=re.findall('[a-z]',a)
b=''
for i,j in zip(all,all_w):
    b+=int(i)*j
print(b)
```

	Input	Expected	Got	
✔	a2b4c6	aabbbbccccccc	aabbbbccccccc	✔
✔	a12b3d4	aaaaaaaaaaaabbbdddd	aaaaaaaaaaaabbbdddd	✔

Passed all tests! ✔

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

Register No.: **Name:**

First N Common Chars

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

Input Format:

The first line contains S1.
The second line contains S2.
The third line contains N.

Output Format:

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

Boundary Conditions:

2 <= N <= 10
2 <= Length of S1, S2 <= 1000

Example Input/Output 1:

Input:

abcbde
cdefghbb
3

Output:

bcd

Note:

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

```
a=input()
b=input()
C=''
d=int(input())
```

```
for i in range(len(a)):
    if(len(C)-d==0):
        break
    else:
        if(a[i]in b):
            if(a[i] not in C):
                C+=a[i]
print (C)
```

	Input	Expected	Got	
✔	abcbde cdefghbb 3	bcd	bcd	✔

Passed all tests! ✔

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

Register No.: **Name:**

Remove Characters

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

Constraints
1<= string length <= 200

Sample Input 1
experience
enc

Sample Output 1
xpri

```
def remove_chars(s1, s2):
    return ''.join([char for char in s1 if char not in s2])
s1=input()
s2=input()
result = remove_chars(s1, s2)
print(result)
```

	Input	Expected	Got	
✔	experience enc	xpri	xpri	✔

Passed all tests! ✔

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

Register No.: **Name:**

Remove Palindrome Words

String should contain only the words are not palindrome.

Sample Input 1
Malayalam is my mother tongue

Sample Output 1
is my mother tongue

For example:

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

```
a=[]
a=input()
b=a. split()
for i in b:
    k=i.lower()
    if k!=k[::-1]:
        print(k,end=' ')
```

	Input	Expected	Got	
✔	Malayalam is my mother tongue	is my mother tongue	is my mother tongue	✔

Passed all tests! ✔

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

Register No.: **Name:**

Return Second World in Uppercase

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

For example:

If input is “Wipro Technologies Bangalore” the function should return “TECHNOLOGIES”
If input is “Hello World” the function should return “WORLD”
If input is “Hello” the program should return “LESS”

NOTE 1: If input is a sentence with less than 2 words, the program should return the word “LESS”.
NOTE 2: The result should have no leading or trailing spaces.

For example:

Input Result
Wipro Technologies Bangalore
TECHNOLOGIES
Hello World
WORLD
Hello
LESS

```
f=input()
s=f.split()
if len(s)>1:
    c=s[1]
    print(c.upper())
else:
    print("LESS")
```

	Input	Expected	Got	
✔	Wipro Technologies Bangalore	TECHNOLOGIES	TECHNOLOGIES	✔
✔	Hello World	WORLD	WORLD	✔
✔	Hello	LESS	LESS	✔

Passed all tests! ✔

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

Register No.: **Name:**

Revers String

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

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

For example:

Input Result
A&x#
x&A#

```
def reverse_string(s):
    s = list(s)
    l, r = 0, len(s) - 1

    while l < r:
        if not s[l].isalpha():
            l += 1
        elif not s[r].isalpha():
            r -= 1
        else:
            s[l], s[r] = s[r], s[l]
            l += 1
            r -= 1

    return ''.join(s)

# Test Cases
print(reverse_string(input())) # Output: "B&A"
```

	Input	Expected	Got	
✔	A&B	B&A	B&A	✔

Passed all tests! ✔

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

Register No.: **Name:**

String characters balance Test

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

For example:

Input Result
Yn
PYnative
True

```
a=input()
b=input()
if a in b:
    print("True")
else:
    print("False")
```

	Input	Expected	Got	
✔	Yn PYnative	True	True	✔
✔	Ynf PYnative	False	False	✔

Passed all tests! ✔

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

Register No.: **Name:**

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

```
a,c=[],[]
for i in range(0,5):
    b=input()
    a.append(b)
for i in range(len(a)):
    if(a[i] not in c):
        c.append(a[i])
        print(a[i])
```

	Input	Expected	Got	
✔	first second first third second	first second third	first second third	✔
✔	rec cse it rec cse	rec cse it	rec cse it	✔

Passed all tests! ✔

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

Register No.: **Name:**

Username Domain Extension

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

Input Format:
The first line contains S.

Output Format:
The first line contains EXTENSION.
The second line contains DOMAIN.
The third line contains USERNAME.

Boundary Condition:
1 <= Length of S <= 100

Example Input/Output 1:

Input:

vijayakumar.r@rajalakshmi.edu.in

Output:

edu.in
rajalakshmi
vijayakumar.r

```
a = input()
ext = a.split('@')[0]
dom = a.split('@')[1].split('.')[0]
userno = a.find('.')
user = a[userno+1:]
print(user)
print(dom, end='\n')
```

print(ext,end='\n')

	Input	Expected	Got	
✔	abcd@gmail.com	com gmail abcd	com gmail abcd	✔

Passed all tests! ✔

07 - Functions

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

Register No.: **Name:**

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

```
def abundant(n):

    l,s=[],0

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

        if(n%i==0):

            l.append(i)
```

```
for i in l:
    s+=i
if(s>n):
    return("Yes")
else:
    return("No")
```

	Test	Expected	Got	
✔	`print(abundant(12))`	Yes	Yes	✔
✔	`print(abundant(13))`	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Automorphic number or not

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

If the number is not valid, it should display “Invalid input”.
If it is an automorphic number display “Automorphic” else display “Not Automorphic”.

Input Format:
Take a Integer from Stdin
Output Format:
Print Automorphic if given number is Automorphic number, otherwise Not Automorphic Example input: 5 Output: Automorphic Example input: 25 Output: Automorphic Example input: 7 Output: Not Automorphic
For example:

Test	Result
print(automorphic(5))	Automorphic

```
def automorphic(n):
    a=str(n*n)
    if(int(a[-1])==n):
        return("Automorphic")
    else:
        return("Not Automorphic")
```

	Test	Expected	Got	
✔	`print(automorphic(5))`	`Automorphic`	`Automorphic`	✔
✔	`print(automorphic(7))`	`Not Automorphic`	`Not Automorphic`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Check Product of Digits

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

Take an input integer from stdin.

Output Format:

Print TRUE or FALSE.

Example Input:

1256

Output:

TRUE

Example Input:

1595

Output:

FALSE

For example:

Test	Result
print(productDigits(1256))	True
print(productDigits(1595))	False

def productDigits(n):

```
a=str(n)
s,p=0,1
for i in range(0,len(a),2):
    s+=int(a[i])
for i in range(1,len(a),2):
    p*=int(a[i])
if(p%s==0):
    return("True")
else:
    return("False")
```

	Test	Expected	Got	
✔	print(productDigits(1256))	True	True	✔
✔	print(productDigits(1595))	False	False	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Christmas Discount

An e-commerce company plans to give their customers a special discount for Christmas. They are planning to offer a flat discount. The discount value is calculated as the sum of all the prime digits in the total bill amount.

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

Constraints

$1 <= orderValue < 10e^{100000}$

Input

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

Output

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

Example Input

578

Output

12

For example:

Test	Result
print(christmasDiscount(578))	12

```
def christmasDiscount(n):
    res=0
    while n!=0:
        rem=n%10
        flag=0
        for i in range(1,rem+1):
            if rem%i==0:
```

```
        flag+=1
    if flag==2:
        res=res+rem
    n=n//10

return res
```

	Test	Expected	Got	
✔	`print(christmasDiscount(578))`	12	12	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

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:

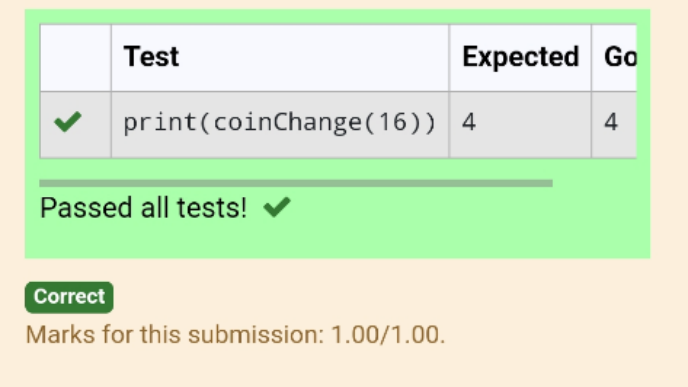
7

Explanation:

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

```
def coinChange(amount):
    # Available coin denominations
    coins = [1, 2, 3, 4]
    # Initialize a list to store the minimum number of coins for each amount from 0 to the target amount
```

```
    dp = [float('inf')] * (amount + 1)
    dp[0] = 0  # Base case: 0 coins needed to make amount 0

    # Iterate through all amounts from 1 to the target amount
    for i in range(1, amount + 1):
        # Iterate through all available coin denominations
        for coin in coins:
            # If the current coin denomination is less than or equal to the current amount
            if coin <= i:
                # Update dp[i] to be the minimum between its current value and dp[i - coin] + 1
                dp[i] = min(dp[i], dp[i - coin] + 1)

    # The result is stored at dp[amount]
    return dp[amount]
    amount = int(input())
    print(coinChange(amount))
```



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

Register No.: **Name:**

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

```
def differenceSum(n):
    a=[]
    b=[]
    k=str(n)
    for i in range(len(k)):
        if int(i)%2==0:
            a.append(int(k[i]))
        else:
            b.append(int(k[i]))
```

s=sum(b)

r=sum(a)

j=s-r

return j

	Test	Expected	Got	
✔	`print(differenceSum(1453))`	1	1	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Ugly number

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

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

```
def checkUgly(n):
    for i in range(n):
    for j in range(n):
    for k in range(n):
        if(n==(2**i)+(3**j)+(5**k)):
        return("ugly")
    return("not ugly")
```

	Test	Expected	Got	
✔	`print(checkUgly(6))`	`ugly`	`ugly`	✔
✔	`print(checkUgly(21))`	`not ugly`	`not ugly`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

08 – Tuple/Set

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

Register No.: **Name:**

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: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

For example:

Input	Result
01010101010	Yes
010101 10101	No

```
a = input()
try:
    c = int(a)
    print("Yes")
except:
    print("No")
```

	Input	Expected	Got	
✔	01010101010	Yes	Yes	✔
✔	REC123	No	No	✔
✔	010101 10101	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

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**.

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

```
t = input()
k = int(input())
a = t.split(",")
l = [int(x) for x in a]
count = 0
x = set()
```

```
for i in range(len(l)):
    for j in range(i + 1, len(l)):
        if l[i] + l[j] == k:
            s = (l[i], l[j])
            if s not in x and (l[j], l[i]) not in x:
                count += 1
                x.add(s)

print(count)
```

	Input	Expected	Got	
✔	5,6,5,7,7,8 13	2	2	✔
✔	1,2,1,2,5 3	1	1	✔
✔	1,2 0	0	0	✔

Passed all tests! ✔

Correct

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

Register No.: **Name:**

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

Example 2:

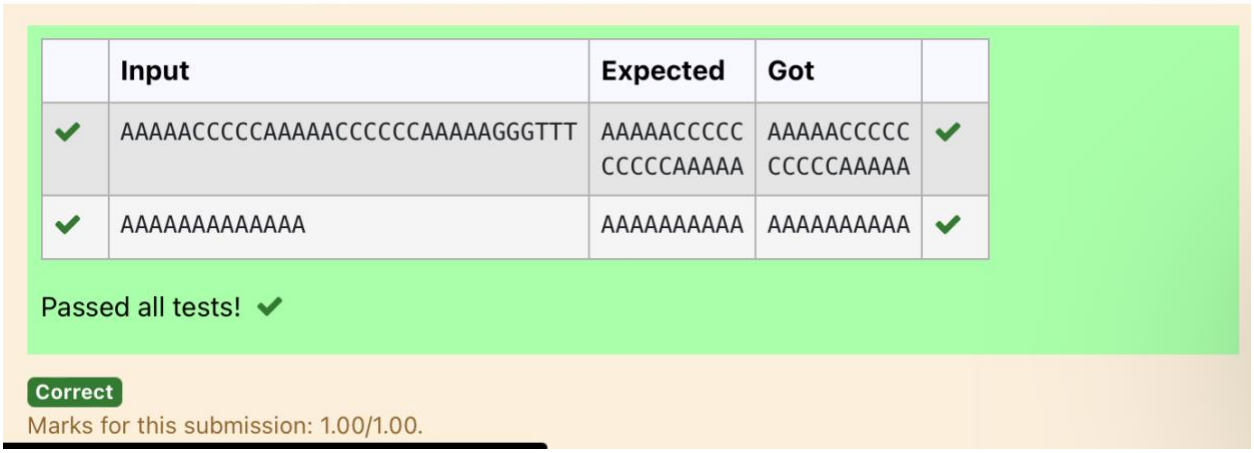
Input: s = "AAAAAAAAAAAAA"
Output: ["AAAAAAAAAA"]

For example:

Input	Result
AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA

```
s = input()
j = []
repeated = set()
for i in range(len(s) - 9):
    sequence = s[i:i+10]
    if sequence in j:
```

```
        repeated.add(sequence)
    else:
        j.append(sequence)
l=list(repeated)
l=list(reversed(l))
for i in l:
    print(i)
```



	Input	Expected	Got	
✔	AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT	AAAAACCCCCCCCCCAAAAA	AAAAACCCCCCCCCCAAAAA	✔
✔	AAAAAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

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.

Example 1:

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

Example 2:

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

For example:

Input	Result
1 3 4 4 2	4

```
n =input().split(" ")
n = list(n)
for i in range(len(n)):
    for j in range(i+1,len(n)):
        if n[i] == n[j]:
            print(n[i])
            exit(0)
```

	Input	Expected	Got	
✔	1 3 4 4 2	4	4	✔
✔	1 2 2 3 4 5 6 7	2	2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

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.

Sample Input:

5 4

1 2 8 6 5

2 6 8 10

Sample Output:

1 5 10

3

Sample Input:

5 5

1 2 3 4 5

1 2 3 4 5

Sample Output:

NO SUCH ELEMENTS

For example:

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

Input	Result
2 6 8 10	

```
a=input()
d=[]
b=input()
c=input()
b=tuple(b.split(" "))
c=tuple(c.split(" "))
for i in b:
    if i not in c:
        d.append(i)
for i in c:
    if i not in b:
        d.append(i)
for i in range(len(d)):
    print(int(d[i]),end=' ')
print()
print(len(d))
```

	Input	Expected	Got	
✔	5 4 1 2 8 6 5 2 6 8 10	1 5 10 3	1 5 10 3	✔
✔	3 3 10 10 10 10 11 12	11 12 2	11 12 2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

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: 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

```
a=input()
b=input()
c=set()
for i in a:
    for j in b:
        if j in i:
            c.add(i)
print(len(c))
```

	Input	Expected	Got	
✔	hello world ad	1	1	✔
✔	Welcome to REC e	1	1	✔
✔	Faculty Upskilling in Python Programming ak	2	2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

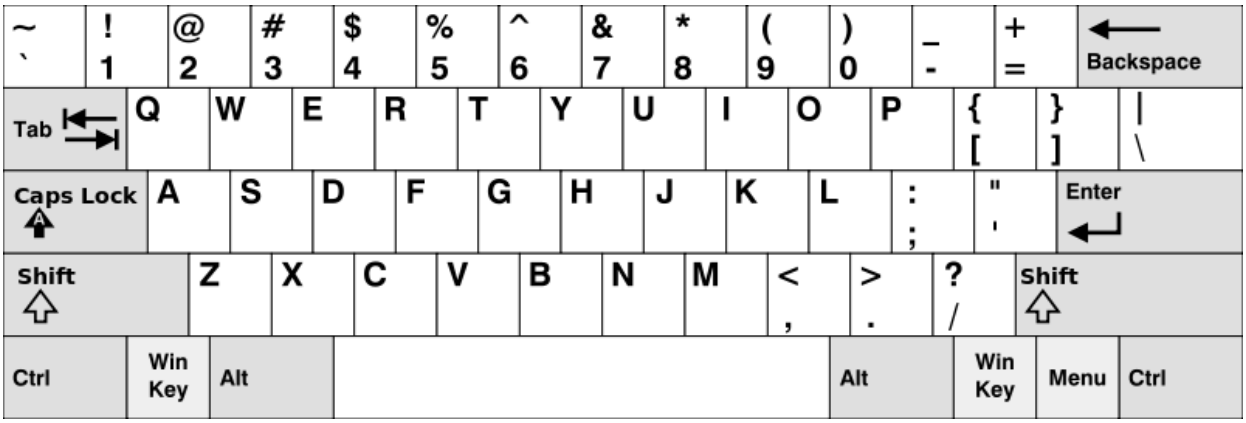
Register No.: **Name:**

American keyboard

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

In the **American keyboard**:

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



Example 1:

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

Example 2:

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

Example 3:

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

For example:

Input	Result
4 Hello	Alaska Dad

Input	Result
Alaska Dad Peace	

```
def findWords(words):
    row1 = set('qwertyuiop')
    row2 = set('asdfghjkl')
    row3 = set('zxcvbnm')

    result = []
    for word in words:
        w = set(word.lower())
        if w.issubset(row1) or w.issubset(row2) or w.issubset(row3):
            result.append(word)
    if len(result) == 0:
        print("No words")
    else:
        for i in result:
            print(i)

a = int(input())
arr = [input() for i in range(a)]
findWords(arr)
```

	Input	Expected	Got	
✔	4 Hello Alaska Dad Peace	Alaska Dad	Alaska Dad	✔
✔	1 omk	No words	No words	✔
✔	2 adsfd afd	adsfd afd	adsfd afd	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

09 – Dictionary

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

Register No.: **Name:**

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.

Example 1:

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

Output: ["sweet","sour"]

Example 2:

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

Output: ["banana"]

Constraints:

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

s1 and s2 consist of lowercase English letters and spaces.

s1 and s2 do not have leading or trailing spaces.

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

Note:

Use dictionary to solve the problem

For example:

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

```
a=input().split()
```

```
b=input().split()
c1,c2,l={},{},[]
for i in a:
    c1[i]=c1.get(i,0)+1
for j in b:
    c2[j]=c2.get(j,0)+1
for w,c in c1.items():
    if(c==1 and w not in b):
        l.append(w)
for w,c in c2.items():
    if(c==1 and w not in a):
        l.append(w)
print(*l)
```

	Input	Expected	Got	
✔	this apple is sweet this apple is sour	sweet sour	sweet sour	✔
✔	apple apple banana	banana	banana	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Sort Dictionary by Values Summation

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

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

Output : {'Gfg': 17, 'best': 18}

Explanation : Sorted by sum, and replaced.

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

Output : {'best': 10, 'Gfg': 16}

Explanation : Sorted by sum, and replaced.

Sample Input:

2

Gfg 6 7 4

Best 7 6 5

Sample Output

Gfg 17

Best 18

For example:

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

```
a=int(input())
d={}
for i in range(a):
```

```
    b=input()
    b=b.partition(" ")
    d[b[0]]=b[-1].split(" ")
n=list(d.values())
k=list(d.keys())
for i in range(len(n)):
    s=0
    for j in range(len(n[i])):
        s+=int(n[i][j])
    d.update({k[i]:s})
l=list(d.items())
if(l[0][1]<l[1][1]):
    for k,v in d.items():
        print(k,v)
else:
    j=1
    for k,v in d.items():
        if(j==1):
            k1,v1=k,v
            j+=1
        else:
            print(k,v)
    print(k1,v1)
```

	Input	Expected	Got	
✔	2 Gfg 6 7 4 Best 7 6 5	Gfg 17 Best 18	Gfg 17 Best 18	✔
✔	2 Gfg 6 6 Best 5 5	Best 10 Gfg 12	Best 10 Gfg 12	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

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.

Examples:

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

Output : John

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

Sample Input:

10
John
John
Johny
Jamie
Jamie
Johny
Jack
Johny
Johny
Jackie

Sample Output:

Johny

For example:

Input	Result
10 John John	Johny

Input	Result
Johny Jamie Jamie Johny Jack Johny Johny Jackie	

```
n = int(input())


votes = {}


for _ in range(n):
    candidate = input()
    votes[candidate] = votes.get(candidate, 0) + 1


max_votes = max(votes.values())
max_candidates = [candidate for candidate, count in
votes.items() if count == max_votes]


print(min(max_candidates))
```

	Input	Expected	Got	
✔	10 John John Johny Jamie Jamie Johny Jack Johny Johny Jackie	Johny	Johny	✔
✔	6 Ida Ida Ida Kiruba Kiruba Kiruba	Ida	Ida	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Student Record

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

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

```
n = int(input())
max_average = float('-inf')
min_average = float('inf')
max_assignment = float('-inf')
min_lab = float('inf')
max_average_students = []
max_assignment_students = []
min_lab_students = []
min_average_students = []
for _ in range(n):
```

```
    name, test, assignment, lab = input().split()
    test = int(test)
    assignment = int(assignment)
    lab = int(lab)
    average = (test + assignment + lab) / 3
    if average > max_average:
        max_average = average
        max_average_students = [name]
    elif average == max_average:
        max_average_students.append(name)
    if average < min_average:
        min_average = average
        min_average_students = [name]
    elif average == min_average:
        min_average_students.append(name)
    if assignment > max_assignment:
        max_assignment = assignment
        max_assignment_students = [name]
    elif assignment == max_assignment:
        max_assignment_students.append(name)
    if lab < min_lab:
        min_lab = lab
        min_lab_students = [name]
    elif lab == min_lab:
        min_lab_students.append(name)
print(*sorted(max_average_students))
print(*sorted(max_assignment_students))
print(*sorted(min_lab_students))
print(*sorted(min_average_students))
```

	Input	Expected	Got	
✔	4 James 67 89 56 Lalith 89 45 45 Ram 89 89 89 Sita 70 70 70	Ram James Ram Lalith Lalith	Ram James Ram Lalith Lalith	✔
✔	3 Raja 95 67 90 Aarav 89 90 90 Shadhana 95 95 91	Shadhana Shadhana Aarav Raja Raja	Shadhana Shadhana Aarav Raja Raja	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Scramble Score

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

The points associated with each letter are shown below:

Points Letters

1 A, E, I, L, N, O, R, S, T and U

2 D and G

3 B, C, M and P

4 F, H, V, W and Y

5 K

8 J and X

10 Q and Z

Sample Input

REC

Sample Output

REC is worth 5 points.

```
letter_scores = {
    'A': 1, 'E': 1, 'I': 1, 'L': 1, 'N': 1, 'O': 1, 'R': 1, 'S': 1, 'T': 1, 'U': 1,
    'D': 2, 'G': 2,
    'B': 3, 'C': 3, 'M': 3, 'P': 3,
    'F': 4, 'H': 4, 'V': 4, 'W': 4, 'Y': 4,
    'K': 5,
```

```
    'J': 8, 'X': 8,
    'Q': 10, 'Z': 10
}
word = input().upper()
score = sum(letter_scores.get(letter, 0) for letter in word)
print(word,"is worth",score,"points.")
```

	Input	Expected	Got	
✔	GOD	GOD is worth 5 points.	GOD is worth 5 points.	✔
✔	REC	REC is worth 5 points.	REC is worth 5 points.	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

10 - Searching & Sorting

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

Register No.: **Name:**

Merge Sort

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

For example:

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

```
a=int(input())
l=[]
l.extend(input().split())
for i in range(a-1):
    for j in range(a-1):
        if(int(l[j])>int(l[j+1])):
            t=int(l[j])
            l[j]=int(l[j+1])
            l[j+1]=t
for i in range(a):
    print(int(l[i]),end=" ")
```

	Input	Expected	Got
✔	5 6 5 4 3 8	3 4 5 6 8	3 4 5 6 8
✔	9 14 46 43 27 57 41 45 21 70	14 21 27 41 43 45 46 57 70	14 21 27 41 43 45 46 5
✔	4 86 43 23 49	23 43 49 86	23 43 49 86

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Bubble Sort

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

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

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

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

Input Format

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

Constraints

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

Output Format

You must print the following three lines of output:

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

Sample Input 0

3

1 2 3

Sample Output 0

List is sorted in 0 swaps.

First Element: 1

Last Element: 3

For example:

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

```
def bubble_sort(arr):
    n = len(arr)
    swaps = 0

    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j + 1]:
                # Swap elements
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
                swaps += 1

    return swaps

# Input the size of the list
n = int(input())

# Input the list of integers
arr = list(map(int, input().split()))

# Perform bubble sort and count the number of swaps
num_swaps = bubble_sort(arr)

# Print the number of swaps
```

```
print("List is sorted in", num_swaps, "swaps.")

# Print the first element
print("First Element:", arr[0])

# Print the last element
print("Last Element:", arr[-1])
```

	Input	Expected	Got	
✔	3 3 2 1	List is sorted in 3 swaps. First Element: 1 Last Element: 3	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	List is sorted in 4 swaps. First Element: 1 Last Element: 9	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Peak Element

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

An element a[i] is a peak element if

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

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

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

Input Format

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

Output Format

Print peak numbers separated by space.

Sample Input

5

8 9 10 2 6

Sample Output

10 6

For example:

Input	Result
4 12 3 6 8	12 8

```
def find_peak(arr):
    peak_elements = []
```

```
    # Check for the first element
    if arr[0] >= arr[1]:
        peak_elements.append(arr[0])

    # Check for middle elements
    for i in range(1, len(arr) - 1):
        if arr[i - 1] <= arr[i] >= arr[i + 1]:
            peak_elements.append(arr[i])

    # Check for the last element
    if arr[-1] >= arr[-2]:
        peak_elements.append(arr[-1])

    return peak_elements

# Input the length of the list
n = int(input())

# Input the list of integers
arr = list(map(int, input().split()))

# Find peak elements and print the result
peak_elements = find_peak(arr)
print(*peak_elements)
```

	Input	Expected	Got	
✔	7 15 7 10 8 9 4 6	15 10 9 6	15 10 9 6	✔
✔	4 12 3 6 8	12 8	12 8	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: **Name:**

Binary Search

Write a Python program for binary search.

For example:

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

```
a = input().split(",")
b = input()
print(b in a)
```

	Input	Expected	Got	
✔	1,2,3,5,8 6	False	False	✔
✔	3,5,9,45,42 42	True	True	✔
✔	52,45,89,43,11 11	True	True	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Register No.: 231501149 **Name:sathyanath**

Frequency of Elements

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

Constraints:

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

Input:

1 68 79 4 90 68 1 4 5

output:

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

For example:

Input	Result
4 3 5 3 4 5	3 2 4 2 5 2

```
def count_frequency(arr):
    frequency = {}

    # Count the frequency of each number in the list
    for num in arr:
        frequency[num] = frequency.get(num, 0) + 1
```

```
    # Sort the dictionary based on keys
    sorted_frequency = sorted(frequency.items())

    # Print the frequency of each number
    for num, freq in sorted_frequency:
        print(num, freq)

# Input the list of numbers
arr = list(map(int, input().split()))

# Count the frequency and print the result
count_frequency(arr)
```

	Input	Expected	Got	
✔	4 3 5 3 4 5	3 2 4 2 5 2	3 2 4 2 5 2	✔
✔	12 4 4 4 2 3 5	2 1 3 1 4 3 5 1 12 1	2 1 3 1 4 3 5 1 12 1	✔
✔	5 4 5 4 6 5 7 3	3 1 4 2 5 3 6 1 7 1	3 1 4 2 5 3 6 1 7 1	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

WEEK 11:

1.

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

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

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

For example:

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

```
try:
    a=input()
    if(len(a)==0):
        print("Error: Please enter a valid age.")
    elif a.isnumeric():
        print("You are",a,"years old.")
    else:
        print("Error: Please enter a valid age.")
except:
    print("Error: Please enter a valid age.")
```

OUTOUT:

	Input	Expected	Got	
	twenty	Error: Please enter a valid age.	Error: Please enter a valid age.	
	25	You are 25 years old.	You are 25 years old.	
	-1	Error: Please enter a valid age.	Error: Please enter a valid age.	
	150	You are 150 years old.	You are 150 years old.	
		Error: Please enter a valid age.	Error: Please enter a valid age.	

Passed all tests!

Correct

2.

Problem Description:

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

Input Format:

A single line input representing the user's age.

Output Format:

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

For example:

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

```
try:
   a=input()
   if(len(a)==0):
      print("Error: Please enter a valid age.")
   elif a.isnumeric():
      print("You are",a,"years old.")
   else:
```

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

OUTPUT:

	Input	Expected	Got	
	25	You are 25 years old.	You are 25 years old.	
	rec	Error: Please enter a valid age.	Error: Please enter a valid age.	
	!@#	Error: Please enter a valid age.	Error: Please enter a valid age.	

Passed all tests!

Correct

3.

Problem Description:

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

Input Format:

User inputs a number.

Output Format:

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

For example:

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

```
def main():
    min_range = 1
    max_range = 100

    try:
```

```
        num = int(input())
        if num < min_range or num > max_range:
            print("Error: Number out of allowed range")
        else:
            print("Valid input.")
    except ValueError:
        print("Error: invalid literal for int()")


if __name__ == "__main__":
```

OUTPUT:

	Input	Expected	Got	
	1	Valid input.	Valid input.	
	100	Valid input.	Valid input.	
	101	Error: Number out of allowed range	Error: Number out of allowed range	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

4.

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

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

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

For example:

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

```
def main():
    try:
        num1 = float(input())
        num2 = float(input())

        division_result = num1 / num2
        modulo_result = num1 % num2

        print(division_result)

    except ValueError:
        print("Error: Non-numeric input provided.")
    except ZeroDivisionError:
        print("Error: Cannot divide or modulo by zero.")



if __name__ == "__main__":
    main()
```

OUTPUT:

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

Passed all tests!

Correct

5.

Problem Description:

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

Input Format:

User inputs a number.

Output Format:

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

For example:

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

```
try:
    a=float(input())
    if(a<0):
        print("Error: Cannot calculate the square root of a negative number.")
    else:
        print("The square root of",a,"is {:.2f}".format(a**0.5))
except:
```

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

OUTPUT:

	Input	Expected	Got	
	16	The square root of 16.0 is 4.00	The square root of 16.0 is 4.00	
	0	The square root of 0.0 is 0.00	The square root of 0.0 is 0.00	
	-4	Error: Cannot calculate the square root of a negative number.	Error: Cannot calculate the square root of a negative number.	

Passed all tests!

Correct

WEEK 12:

1.

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

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

Problem Statement

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

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

Input Format

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

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

The third line contains an integer, k.

Output Format

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

Constraints:

$1 \leq n \leq 10^5$

$-10^4 \leq nums[i] \leq 10^4$

$0 \leq k \leq 10^4$

For example:

Input	Result
5 1 3 1 5 4 0	1
4 1 2 2 1 1	4

```
def count_pairs_with_difference_k(activities, k):
    count = 0
    n = len(activities)
    for i in range(n):
        for j in range(i + 1, n):
            if abs(activities[i] - activities[j]) == k:
                count += 1
```

```
    return count


# Reading input
n = int(input())
activities = list(map(int, input().split()))
k = int(input())

# Calling function and printing the result
print(count_pairs_)
```

OUTPUT:

	Input	Expected	Got	
	4 1 2 3 4 1	3	3	
	5 1 3 1 5 4 0	1	1	
	4 1 2 2 1 1	4	4	

Passed all tests!

Correct

2.

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

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

For example:

Input	Result
16	True
5	False

```
def is_power_of_four(n):
    if n <= 0:
        return False
    while n > 1:
        if n % 4 != 0:
            return False
        n //= 4
    return True


# Test the function
n = int(input())
print(is_power_of_four(n))
```

OUTPUT:

	Input	Expected	Got	
	16	True	True	
	5	False	False	
	1	True	True	
	-1	False	False	

Passed all tests!

Correct

3. Background:

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

Problem Statement:

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

Input Format:

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

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

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

Output Format:

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

Constraints:

$1 \leq N \leq 100$

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

Marks will be non-negative integers.

For example:

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

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

```
def calculate_average_marks(data):
    total_marks = 0
    num_students = 0

    for student in data:
        if 'MARKS' in student:
            total_marks += int(student['MARKS'])
            num_students += 1

    if num_students == 0:
        return 0

    return total_marks / num_students

# Read input
N = int(input())
columns = input().split()

# Initialize data structure to store student records
students = []

# Read student data
for _ in range(N):
    student_data = input().split()
```

```
    student_record = {columns[i]: student_data[i] for i in range(len(columns))}
    students.append(student_record)

# Calculate average marks
average_marks = calculate_average_marks(students)

# Print average marks with two decimal places
print("{:.2f}".format(average_marks))
```

OUTPUT:

	Input	Expected	Got	
	3 ID NAME MARKS CLASS 101 John 78 Science 102 Doe 85 Math 103 Smith 90 History	84.33	84.33	
	3 MARKS CLASS NAME ID 78 Science John 101 85 Math Doe 102 90 History Smith 103	84.33	84.33	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

4.

Background:

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

Problem Statement:

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

Input Format:

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

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

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

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

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

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

Output Format:

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

Constraints:

$1 \leq X \leq 1000$ — Raghu's shop can hold between 1 and 1000 shoes.

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

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

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

For example:

Input	Result
10 2 3 4 5 6 8 7 6 5 18 6 6 55 6 45	200

Input	Result
6 55 4 40 18 60 10 50	
5 5 5 5 5 5 5 5 10 5 10 5 10 5 10 5 10	50

Answer:(penalty regime: 0 %)

```
class ShoeInventory:
    def __init__(self, shoe_sizes):
        self.inventory = {size: 0 for size in shoe_sizes}

    def add_shoes(self, size, quantity):
        self.inventory[size] += quantity

    def sell_shoes(self, size):
        if self.inventory.get(size, 0) > 0:
            self.inventory[size] -= 1
            return True
        return False

class TransactionManager:
    def __init__(self, shoe_inventory):
        self.shoe_inventory = shoe_inventory
        self.total_revenue = 0

    def process_transactions(self, customer_requests):
```

```
        for size, price in customer_requests:
            if self.shoe_inventory.sell_shoes(size):
                self.total_revenue += price

# Read input
X = int(input())  # Total number of shoes
shoe_sizes = list(map(int, input().split()))  # Shoe sizes available
N = int(input())  # Number of customer requests

# Initialize shoe inventory
inventory = ShoeInventory(shoe_sizes)

# Populate initial inventory
for size in shoe_sizes:
    inventory.add_shoes(size, X // len(shoe_sizes))

# Initialize transaction manager
transaction_manager = TransactionManager(inventory)

# Process transactions
for _ in range(N):
    size, price = map(int, input().split())
    transaction_manager.process_transactions([(size, price)])

# Print total revenue
print(transaction_manager.total_revenue)
```

OUTPUT:

	Input	Expected	Got	
	10 2 3 4 5 6 8 7 6 5 18 6 6 55 6 45 6 55 4 40 18 60 10 50	200	200	
	5 5 5 5 5 5 5 5 10 5 10 5 10 5 10 5 10	50	50	
	4 4 4 6 6 5 4 25 4 25 6 30 6 55 6 55	135	135	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

5.

Background:

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

Problem Statement:

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

Input Format:

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

Input terminates with a blank line.

Output Format:

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

Constraints:

Book titles and genres are strings.

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

Genres will not exceed 50 characters.

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

For example:

Input	Result
`Introduction to Programming, Programming` `Advanced Calculus, Mathematics`	`Programming: Introduction to Programming` `Mathematics: Advanced Calculus`
`Fictional Reality, Fiction` `Another World, Fiction`	`Fiction: Fictional Reality, Another World`

```
def categorize_books_sorted():
    books = {}

    while True:
        try:
            user_input = input().strip()
        except EOFError:
            break
```

```
    if not user_input:
        break

    title, genre = user_input.split(',')
    genre = genre.strip()

    if genre in books:
        books[genre].append(title)
    else:
        books[genre] = [title]

for genre in sorted(books.keys()):
    print(f"{genre}: {', '.join(books[genre])}")

categorize_books_sorted()
```

OUTPUT:

	Input	Expected	Got	
	Introduction to Programming, Programming Advanced Calculus, Mathematics	Programming: Introduction to Programming Mathematics: Advanced Calculus	Programming: Introduction to Programming Mathematics: Advanced Calculus	
	Fictional Reality, Fiction Another World, Fiction	Fiction: Fictional Reality, Another World	Fiction: Fictional Reality, Another World	

Passed all tests!

Correct