WEEK - 1
Variables, Datatypes in Python

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

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

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

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

Input	Result
197	9
-197	9
5	-1

```
n = int(input())
s = str(n)
if (len(s) == 1):
    print(-1)
else:
    print(s[len(s)-2])
```

2)In a Lab 36% are Dell and 34% Lennovo and 28% are Acer and 2% are Samsung. write a python code to print total systems and brand wise count in the specific format using sep operator.

Input	Result
150	Total System:150 Dell:54 Lennovo:51

	Acer:42 Samsung:3

```
n = int(input())
print("Total System:",n, sep='')
print("Dell:",int(n * 0.36), sep='')
print("Lennovo:",int(n * 0.34), sep='')
print("Acer:",int(n * 0.28), sep='')
print("Samsung:",int(n * 0.02), sep='')
```

3) 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.

For example:

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

```
a = int(input())
b = int(input())
c = a * 0.10 + b * 0.25
print("Your total refund will be $%.2f."%(c))
```

4) 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

For example:

Input	Result
45500 500 60000	30.43 is the gain percent.

```
x = int(input())
y = int(input())
z = int(input())
ans = (z - (x + y)) / (x + y) * 100
print("%.2f is the gain percent." %(ans))
```

5) You went on a tour to Ooty with your friends. As a part of the tour, you went boating with them. For the boat to remain stable, the number of people on one boat is restricted based on the weight of the people. You find that the boatman who is sailing your boat is so much greedy of money. For earning more, he takes too many people to travel in the boat at a time. So you want to check how many people can travel in the boat at a time so that the boat will not drown. Calculate the weight by considering the number of adults and number of children. Assume that an adult weighs 75 kg and children weigh 30 kg each. If the weight is normal, display Boat is stable, else display Boat will drown.

Input Format:

Input consists of 3 integers. First input corresponds to the weight that the boat can handle. Second input corresponds to the number of adults. Third input corresponds to the number of children.

Input	Result
340	Boat is stable

2 3	

Code:

```
bw = int(input())
aw = int(input())
cw = int(input())
if ((aw * 75) + (cw * 30)) > bw:
    print("Boat will drow")
else:
    print("Boat is stable")
```

6) In a Logistic the Parcels to be delivered in 4 locations (1st locaion 20%, 2nd location 40%, 3rd location 30% and 4th location 10%). write a python code to find the total no. of parcels after the delivery in 2 locations . use a format() to print the no of parcels delivered in in each location

Input	Result
250	Total Parcels is 250 1st Location 50 parcels 2nd Location 100 parcels 3rd Location 75 parcels 4th Location 25 parcels

```
n = int(input())
print("Total Parcels is",n)
print("1st Location",int(n * 0.20),"parcels")
print("2nd Location",int(n * 0.40),"parcels")
print("3rd Location",int(n * 0.30),"parcels")
print("4th Location",int(n * 0.10),"parcels")
```

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

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

```
a = int(input())
b = float(input())
print(a,',',type(a), sep='')
print(round(b, 1),',',type(b), sep='')
```

8) 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.

For example:

Input	Result
10000	16000

```
n = int(input())
da = n * 0.40
hr = n * 0.20
print(int(n + da + hr))
```

9) In department 54% are boys and 46% are girls and 8% are hostel (boys/girls). write a python code to print total no of boys, girls and hostel students in the specific format using modulo operator.

Input	Result
1500	Total Students : 1500, Boys : 810, Girls : 690, Hostel : 120

```
n = int(input())
print("Total Students :", n, end = ", ")
print("Boys :", int(n * 0.54), end = ', ')
print("Girls :", int(n * 0.46), end = ', ')
print("Hostel :", int(n * 0.08))
```

10) 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.

Input	Result
450	weekdays 10.38 weekend 0.38

```
n = int(input())
a = abs((n - 500) / 130)
print("weekdays %.2f"%(a+10))
print("weekend %.2f"%(a))
```

WEEK - 2

Operators and Formatting Output

1) Rohit wants to add the last digits of two given numbers. Write a program.

Input	Result
267 154	11
267 -154	11

```
a, b = int(input()), int(input())
print(abs(a) % 10 + abs(b) % 10)
```

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

Input	Result
197	7
-197	7

3) Complete the program to convert days into years, month and days. (Ignoring leap year and considering 1 month is 30 days)

Input	Result
375	YEARS: 1 MONTH: 0 DAYS: 10

```
n = int(input())
if (n < 365):
    y = 0
    m = n // 30
    d = n % 30
    print("YEARS:",y,"MONTH:", m,"DAYS:", d)

else:
    m = n // 365
    r = n % 365
    if (r < 30):
```

```
    print("YEARS:",m,"MONTH:", 0,"DAYS:", r)
else:
    print("YEARS:",m, "MONTH:", r // 30, "DAYS:", r % 30)
```

4) 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 consists of two integers that correspond to the age and weight of a person respectively.Display True(IF ELIGIBLE)Display False (if not eligible)

Input	Result
18 40	False

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

5) 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

Input	Result
197	7
-197	7

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

6) 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.

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.")
```

7) Write a program to find whether the given input number is Even.

If the given number is even, the function should return 2 else it should return 1.

Note: The number passed to the program can either be negative, positive or zero. Zero should be treated as Even.

Input	Result
100	2
1001	1

```
n = int(input())
if (abs(n) % 2 == 0):
    print(2)
else:
    print(1)
```

8) 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.

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

Input	Result
3	2

```
n = int(input())
c = 0
a = bin(n)
for i in a:
    if (i == '1'):
        c += 1
print(c)
```

9) 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.

Input	Result
32 43	False

```
w, s = int(input()), int(input())
if (w % 3 == 0 and s % 2 == 0):
    print("True")
else:
    print("False")
```

10) 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	Result
0	C

```
n = int(input())
if (n == 1):
    print("D")
else:
    print("C")
```

WEEK - 3
Selection Controls

1) 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 at least 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”.

Input	Result
8 3	OUT

```
a, b = int(input()), int(input())
if (b < a // 2):
    print("OUT")
else:
```

```
    print("IN")
```

2) 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.

Input	Result
February	February has 28 or 29 days in it.
March	March has 31 days in it.
April	April has 30 days in it.

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

```
    print("July has 31 days in it.")
elif (m == "August"):
    print("August has 30 days in it.")
elif (m == "September"):
    print("Septamber has 31 days in it.")
elif (m == "October"):
    print("October has 30 days in it.")
elif (m == "November"):
    print("November has 31 days in it.")
else:
    print("December has 30 days in it.")
```

3) 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

Input	Result
70 60 80	The candidate is eligible

```
a, b, c = int(input()), int(input()), 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")
```

4) 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.

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

```
a = input()
if (a in ('a', 'e', 'i', 'o', '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.")
```

5) 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.

Input	Result
1900	1900 is not a leap year.

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

6) Write a Python program that accepts three parameters. The first parameter is an integer. The second is one of the following mathematical operators: +, -, /, or *. The third parameter will also be an integer.

The function should perform a calculation and return the results. For example, if the function is passed 6 and 4, it should return 24.

Input	Result
11 + 14	25

```
a, b, c = int(input()), input(), int(input())
if (b == "+"):
    print(a + c)
elif (b == "-"):
    print(a - c)
elif (b == "*"):
    print(a * c)
elif (b == "/"):
    print(a/ c)
```

7) 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 multiples 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.

Input	Result
32 43	False

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

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

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

Input	Result
60 60 60	That's a equilateral triangle
40 40 80	That's a isosceles triangle

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

9) 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/-

Input	Result
100.00	120.00

```
a = float(input())
amt = 0
if (a <= 199):
    amt = a * 1.20
elif (a >= 200 and a < 400):
    amt = a * 1.50
elif (a >= 400 and a < 600):
    amt  = a * 1.80
elif (a >= 600):
    amt = a * 2


if (amt < 100):
    print("100.00")
elif (amt >= 100 and amt <= 400):
    print("%.2f"%amt)
else:
    print("%.2f"%(amt + (amt * 0.15)))
```

10) Write a program that reads an integer from the user. Then your program should display a message indicating whether the integer is even or odd.

Input	Result
5	5 is odd

```
a = int(input())
if (a % 2 == 0):
    print(a, "is even.")
else:
    print(a, "is odd.")
```

WEEK - 4

Iteration Controls

1) 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)

Input: 4

Output: 1234

Explanation: As input is 4, We have to take 4 terms. 1 + 11 + 111 + 1111

Input	Result
3	123

```
n = int(input())
s = 0
for i in range(1, n+1):
    s += int('1' * i)
print(s)
```

2) Write a program to find the count of ALL 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.

Input	Result
293	3

```
n = int(input())
s = str(n)
print(len(s))
```

3) A number is stable if each digit occur the same number of times.i.e, the frequency of each digit in the number is the same. For e.g. 2277,4004,11,23,583835,1010 are examples for stable numbers.
Similarly, a number is unstable if the frequency of each digit in the number is NOT same.

Input	Result
2277	Stable Number

```
n = int(input())
s = str(n)
a = list(set(s))
arr = []
for i in range(len(a)):
    arr.append(s.count(a[i]))
f = arr[0]
z = 0
for i in range(len(arr)):
```

```
    if (arr[i] != f):
        z = 1
if (z == 1):
    print("Unstable Number")
else:
    print("Stable Number")
```

4) Write a program that reads a positive integer, n, from the user and then displays the sum of all of the integers from 1 to n.

Input	Result
10	The sum of the first 10 positive integers is 55.0

```
n = int(input())
print("The sum of the first",n,"positive integers is",(n * (n + 1)) / 2)
```

5) 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.

Input	Result
8	13

```
n = int(input())
if (n == 0):
```

```
    print(0)
elif (n == 1):
    print(1)
else:
    a = 0
    b = 1
    for i in range(2, n):
        t = a + b
        a = b
        b = t
    print(b)
```

6) 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)

Input: 4

Output: 1234

Explanation:as input is 4, have to take 4 terms. 1 + 11 + 111 + 1111

```
n = int(input())
s = 0
for i in range(1, n+1):
    s += int("1" * i)
print(s)
```

Input	Result
3	123

7) Strong number is a special number whose sum of factorial of digits is equal to the original number.

For example: 145 is strong number. Since, 1! + 4! + 5! = 145.

Write a program to find whether the given number is a Strong Number or not.

Input	Result
145	Yes

```
import math
n = int(input())
s = 0
for i in str(n):
    s += math.factorial(int(i))
if (s == n):
    print("Yes")
else:
    print("No")
```

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

Input	Result
20	1 2 4 5 10 20

```
n = int(input())
for i in range(1, n+1):
    if (n % i == 0):
        print(i, end=' ')
```

9) You are choreographing a circus show with various animals. For one act, you are given two kangaroos on a number line ready to jump in the positive direction.

•The first kangaroo starts at position x1 and moves at a speed v1 meters per jump.

•The second kangaroo starts at position x2 and moves at a speed of v2 meters per jump and x2 > x1

•You have to figure out how to get both kangaroos at the same position at the same time as part of the show before k jumps. If it is possible, return YES, otherwise return NO.

Input Format:

x1-position of kangaroo1

v1-Speed of kangaroo1

x2-position of kangaroo2

v2-Speed of kangaroo2

k-jumps

Output Format:

Both kangaroos are at the same position within k jumps, YES, otherwise NO.

Input	Result
0 3 4 2 6	YES

```
x1 = int(input())
v1 = int(input())
x2 = int(input())
v2 = int(input())
k = int(input())
i = 1
s = 0
while (i < k):
    x1 += v1
    x2 += v2
    if (x1 == x2):
        s = 1
```

```
        break
    i += 1
if (s == 1):
    print("YES")
else:
    print("NO")
```

10) Write a program to check whether a given number is a perfect number or not.

Perfect number is a positive number whose sum of all positive divisors excluding that number is equal to that number.

For example, 6 is the perfect number since divisors of 6 are 1, 2 and 3.

Sum of its divisor is 1 + 2+ 3 = 6

Input	Result
6	YES

```
n = int(input())
s = 0
for i in range(1, n):
    if (n % i == 0):
        s += i
if (s == n):
    print("YES")
else:
    print("NO")
```

WEEK - 5

Functions - User Defined

1) 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.

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.

Test	Result
print(christmasDiscount(578))	12

```
def christmasDiscount(n):
    sum1 = 0
    s = str(n)
    for i in s:
        k = 0
        for j in range(2, int(i)):
            if (int(i) % j == 0):
                k = 1
                break
        if (k == 0):
            sum1 += int(i)
        k = 0
    return sum1
```

2) Write a function that returns the value of a+aa+aaa+aaaa with a given digit as the value of a.Suppose the following input is supplied to the program: 9 Then, the output should be:9+99+999+9999=11106

Test	Result
print(Summation(8))	9872

```
def Summation(n):
    if (n == 10):
```

```
    return 10203040
  return n + (n * 11) + (n * 111) + (n * 1111)
```

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

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,

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

```
def checkUgly(n):
  t = n
  for i in (2, n):
    if (n % i == 0):
      flag = 1
      break
  if (flag == 1 and (i == 2 or i == 3 or i == 5)):
    return "ugly"
  else:
    return "not ugly"
```

4) A strobogrammatic number is a number that looks the same when rotated 180 degrees (looked at upside down). Write a program to determine if a number is strobogrammatic. The number is represented as a string.

Test	Result

print(Strobogrammatic(69))	true
print(Strobogrammatic(962))	false

```
def Strobogrammatic(n):
    s = str(n)
    d = s[::-1]
    s1 = ''
    for i in d:
        if (i == '6'):
            s1 += '9'
        elif (i == '9'):
            s1 += '6'
        else:
            s1 += i
    if (str(n) == s1):
        return "true"
    else:
        return "false"
```

5) 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.

Test	Result
16	4

25	7

```
def coinChange(n):
    t = 0
    c = 0
    k = 0
    for i in range(4, 0, -1):
        while(i > 0):
            t += i
            c += 1
            if (t > n):
                t -= i
                c-=1
                break
            if (t == n):
                return c
```

WEEK - 5

Functions - Recursion

1) Complete a Recursive Function to find if a given number N can be expressed as a sum of two prime numbers.

Note: YOU MUST OPTIMIZE the logic to find whether a number is prime or not, as very large prime numbers are provided as input. If the logic is not optimized your program will NOT get executed within the given time limit.

Test	Result

print(checkPrimeSum(20))	yes
print(checkPrimeSum(23))	no

```
def checkPrimeSum(n):
    arr = []
    k = 0
    for i in range(2, n):
        for j in range(2, i):
            if (i % j == 0):
                k = 1
                break
        if (k == 0):
            k = 0
            arr.append(i)
        else:
            k = 0
    k = 0
    for i in range(len(arr)):
        for j in range(i+1, len(arr)):
            if (arr[i] + arr[j] == n):
                k = 1
                break
        if (k == 1):
            return "yes"
    if (k == 0):
        return "no"
```

2) Given an integer number, you have to count the digits using recursion using a Python program. In this program, you will be reading an integer number and counting the total digits, using a function countDigits() which will take a number as an argument and return the count after recursion process..

Test	Result
print(countDigits(800))	3

```
def countDigits(n):
    s = str(n)
    return len(s)
```

3) Write a program that reads a string from the user and uses your recursive function to determine whether or not it is a palindrome. Then your program should display an appropriate message for the user.

Test	Result
malayalam	That was a palindrome!

```
def isPalindrome(s):
    st = 0
    end = len(s) - 1
    k = 0
    while(st <= end):
        if (s[st] != s[end]):
            k = 1
            break
        else:
            st += 1
            end -= 1
    if (k == 0):
        print("That was a palindrome!")
```

```
    else:
        print("That is not a palindrome.")

s = input()
isPalindrome(s)
```

4) Euclid was a Greek mathematician who lived approximately 2,300 years ago. His algorithm for computing the greatest common divisor of two positive integers, a and b, is both efficient and recursive. It is outlined below:

If b is 0 then
return a
else
Set c equal to the remainder when a is divided by b
Return the greatest common divisor of b and c

Write a Recursive funtion that implements Euclid's algorithm and uses it to determine thegreatest common divisor of two integers entered by the user. Test your program with some very large integers. The result will be computed quickly, even for huge numbers consisting of hundreds of digits, because Euclid's algorithm is extremely efficient.

Test	Result
print(gcd(8,12))	4
print(gcd(8,12))	40

```
def gcd(a,b):
    if (b == 0):
        return a
    else:
        c = a % b
        return gcd(b, c)
```

5) Complete the recursive function to return the Binary Equivalent of an Integer using Recursion.

Test	Result

print(binayNumber(10))	1010
print(binayNumber(257))	100000001

```
def binayNumber(n):
    b = bin(n)
    return b[2:]
```

WEEK - 6

Strings

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

Test	Result
thistest123string 123	8

```
s1 = input()
s2 = input()
print(s1.find(s2))
```

2) Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.An input string is valid if Open brackets must be closed by the same type of brackets & Open brackets must be closed in the correct order.

s consists of parentheses only '()[]{}'.

Test	Result
print(ValidParenthesis("()"))	true
print(ValidParenthesis("()[]{}"))	true

print(ValidParenthesis("(]"))	false

```
def ValidParenthesis(s):
    arr = []
    for i in s:
        if (i == "(" or i == "{" or i == "["):
            arr.append(i)
        else:
            if (i == ")"):
                if ("(" in arr):
                    arr.pop()
            elif (i == "}"):
                if ("{" in arr):
                    arr.pop()
            else:
                if ("[" in arr):
                    arr.pop()
    if (len(arr) == 0):
        return "true"
    else:
        return "false"
```

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

Test	Result
a2b4c6	aabbbbccccccc

```
s = input()
s1 = ""
t = ""
n = 0

for i in range(len(s)):
    if (s[i].isalpha()):
        s1 += t * n
        n = 0
```

```
        t = s[i]
    else:
        n = (n * 10) + int(s[i])
s1 += t * n
print(s1)
```

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

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

Input format:

Take string as an input from stdin.

Output format:

Print the word is key word or not.

Input	Result
break	break is a keyword
IF	IF is not a keyword

```
s = input()
if (s in ("break", "case", "continue", "defer", "else", "for", "func", "goto", "if", "map", "range", "type", "var")):
    print(s, "is a keyword")
else:
    print(s, "is not a keyword")
```

5) Given a non-empty string s and an abbreviation abbr, return whether the string matches with the given abbreviation.

The string "word" contains only the following valid abbreviations: ["word", "1ord", "w1rd", "wo1d", "wor1", "2rd", "w2d", "wo2", "1o1d", "1or1", "w1r1", "1o2", "2r1", "3d", "w3", "4"]

Notice that only the above abbreviations are valid abbreviations of the string "word". Any other string is not a valid abbreviation of "word".

Note:Assume s contains only lowercase letters and abbr contains only lowercase letters and digits.

Test	Result
internationalization i12iz4n	true
apple a2e	false

```
s = input()
a = input()
n = len(s)
m = len(a)

i, j = 0, 0
while i < n and j < m:
    if (a[j].isdigit()):
        num = 0
        while j < m and a[j].isdigit():
            num =  num * 10 + int(a[j])
            j += 1
        i += num
    else:
        i += 1
        j += 1

if (i == n and j == m):
    print("true")
else:
    print("false")
```

6) Write a Python program to get one string and reverse a string. The input string is given as an array of characters char[].You may assume all the characters consist of printable ascii characters.

Test	Result
hello	olleh
Hannah	hannaH

```
s = input()
print(s[::-1])
```

7) A pangram is a sentence where every letter of the English alphabet appears at least once.Given a string sentence containing only lowercase English letters, return true if sentence is a pangram, or false otherwise.

Test	Result
print(checkPangram('thequickbrownfoxjumpsoverthelazydog'))	true

```
def checkPangram(s):
    k = 0
    for i in range(97, 123):
        if (chr(i) not in s):
            k = -1
            break
    if (k == -1):
        return "false"
```

```
    else:
        return "true"
```

8) 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	Result
arvijayakumar@rajalakshmi.edu.in	edu.in rajalakshmi arvijayakumar

```
s = input()
i1 = 0
i2 = 0
for i in range(len(s)):
    if (s[i] == "@"):
        i1 = i
        break
for i in range(i1 + 1, len(s)):
    if (s[i] == "."):
        i2 = i
        break

for i in range(i2 + 1, len(s)):
    print(s[i], end="")
print()
for i in range(i1 + 1, i2):
    print(s[i], end="")
print()
for i in range(i1):
    print(s[i], end="")
```

9) Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

Note: For the purpose of this problem, we define empty string as valid palindrome.

Input	Result
A man, a plan, a canal: Panama	1
race a car	0

```
s = input()
s1 = ""
for i in s:
    if (i.isalpha()):
        s1 += i
r = s1[::-1]
s1 = s1.lower()
r = r.lower()
if (s1 == r):
    print(1)
else:
    print(0)
```

10) The program must accept the N series of keystrokes as string values as the input. The character ^ represents undo action to clear the last entered keystroke. The program must print the string typed after applying the undo operations as the output. If there are no characters in the string then print -1 as the output.

Input	Result
3 Hey ^ goooo^^glee^ lucke^y ^charr^ms ora^^nge^^^^	Hey google luckycharms -1

```
n = int(input())
s1 = ""
```

```
for i in range(n):
    s = input()
    for i in s:
        if (i != "^"):
            s1 += i
        else:
            s1 = s1[:len(s1) - 1]
    if (len(s1) == 0):
        print(-1)
    else:
        print(s1)
    s1 = ""
```

WEEK - 7

Lists

1) An array is monotonic if it is either monotone increasing or monotone decreasing.
An array A is monotone increasing if for all i <= j, A[i] <= A[j]. An array A is monotone decreasing if for all i <= j, A[i] >= A[j].

Write a program if n array is monotonic or not. Print "True" if is monotonic or "False" if it is not. Array can be monotone increasing or decreasing.

Input	Result
4 6 5 4 3	True

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

```
    a.append(int(input()))
x,f=0,0
for i in range(1,n):
    if a[i]>a[i-1]:
        x+=1
    elif a[i]<a[i-1]:
        f+=1
if x==0 or f==0:
    print(True)
else:
    print(False)
```

2) Given two arrays of positive integers, for each element in the second array, find the total number of elements in the first array which are *less than or equal to* that element. Store the values determined in an array.

For example, if the first array is *[1, 2, 3]* and the second array is *[2, 4]*, then there are *2* elements in the first array *less than or equal to 2*. There are *3* elements in the first array which are *less than or equal to 4*. We can store these answers in an array, *answer = [2, 3]*.

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

```
n=int(input())
a=[]
for i in range(n):
    a.append(int(input()))
m=int(input())
b=[]
for i in range(m):
```

```
    b.append(int(input()))
ans=[]
for j in range(m):
    c=0
    for i in range(n):
        if a[i]<=b[j]:
            c+=1
    ans.append(c)
for i in range(len(ans)):
    print(ans[i])
```

3) Program to print all the distinct elements in an 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

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

```
n = int(input())
```

```
arr = []
for i in range(n):
    a = int(input())
    arr.append(a)

s = set(arr)
for i in s:
    print(i, end=" ")
```

4) Given an integer n, return an list of length n + 1 such that for each i (0 <= i <= n), ans[i] is the number of 1's in the binary representation of i.

Test	Result
print(CountingBits(5))	[0, 1, 1, 2, 1, 2]

```
def CountingBits(n):
    a=[]
    b=0
    for i in range(n+1):
        b=bin(i)
        a.append((b.count('1')))
    return a
```

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

Eg,: n = 20 p = 3

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

Input	Result
10 3	5

10 5	0
1 1	1

```
n=int(input())
p=int(input())
a=[]
c=0
for i in range(1,n+1):
    if n % i==0:
        a.append(i)
        c+=1
if p<=len(a):
    print(a[p-1])
else:
    print(0)
```

6) The program must accept N integers and an integer K as the input. The program must print every K integers in descending order as the output.

Note: If N % K != 0, then sort the final N%K integers in descending order.

Input Format:

The first line contains the values of N and K separated by a space.

The second line contains N integers separated by space(s).

Output Format:

The first line contains N integers.

Input	Result

7 3 48 541 23 68 13 41 6	541 48 23 68 41 13 6

```
n, k = map(int , input().split())
arr = list(map(int, input().split()))
b = []
for i in range(0, n, k):
    b.extend(sorted(arr[i : i + k], reverse = True))
print(*b)
```

7) 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	Result
1 3 1 3 5 4	1
1 3 1 3 5 99	0

```
t=int(input())
while(t):
    n=int(input())
    a=[]
    c=0
    for i in range(n):
        a.append(int(input()))
```

```
    k=int(input())
    for i in range(n):
        for j in range(n):
            if a[i]-a[j]==k and i!=j:
                c+=1
    if c:
        print(1)
    else:
        print(0)
    t-=1
```

8) Given a matrix mat where every row is sorted in strictly increasing order, return the smallest common element in all rows.

If there is no common element, return -1.

Input	Result
4 5 1 2 3 4 5 2 4 5 8 10 3 5 7 9 11 1 3 5 7 9	5

```
x=input().split()
n,m=int(x[0]),int(x[1])
ans=[]
for i in range(n):
    x=input().split()
    a=[]
    for j in x:
        a.append(int(j))
```

```
    ans.append(a)
inter=set(ans[0])
for i in range(1,n):
    inter=inter & set(ans[i])
print(min(list(inter)))
```

9) Assume you have an array of length *n* initialized with all 0's and are given *k* update operations.

Each operation is represented as a triplet: [startIndex, endIndex, inc] which increments each element of subarray A[startIndex ... endIndex] (startIndex and endIndex inclusive) with inc.

Return the modified array after all *k* operations were executed.

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

```
n=int(input())
a=[]
for i in range(n):
    a.append(0)
m=int(input())
for i in range(m):
    x=input().split()

    sta,sto,inc=int(x[0]),int(x[1]),int(x[2])
    for j in range(int(sta),int(sto)+1):
        a[j]+=inc
print(*a)
```

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

Input	Result
7 23 45 23 56 45 23 40	23 occurs 3 times 45 occurs 2 times 56 occurs 1 times 40 occurs 1 times

```
n=int(input())
a=[]
b=[]
c=[]
for i in range(n):
    a.append(int(input()))
for i in range(n):
    if a[i] not in b:
        c.append(a.count(a[i]))
        b.append(a[i])
for i in range(len(c)):
    print(f"{b[i]} occurs {c[i]} times")
```

WEEK - 8

Tuples & Sets

1) 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.

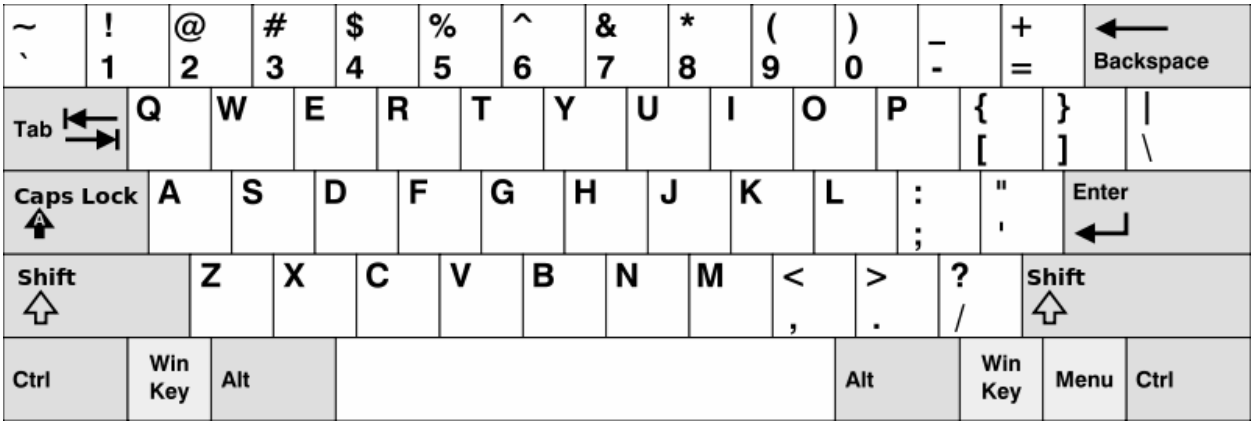
Input	Result
01010101010	Yes
REC101	No

```
s = input()
k = 0
for i in s:
    if (i == '0' or i == '1'):
        continue
    else:
        k = 1
        break
if (k == 1):
    print("No")
else:
    print("Yes")
```

2) Given an array of string words, return the words that can be typed using letters of the alphabet on only one row of the 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".



Input	Result
["Hello","Alaska","Dad","Peace"]	["Alaska","Dad"]
["omk"]	[]

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

r1 = "qwertyuiop"
r2 = "asdfghjkl"
r3 = "zxcvbnm"


c = 0
for i in range(len(arr)):
    s = set(arr[i].lower())
    if (s.issubset(r1) or s.issubset(r2) or s.issubset(r3)):
        print(arr[i])
        c += 1

if (c == 0):
    print("No words")
```

3) 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	Result
5 4 1 2 8 6 5 2 6 8 10	1 5 10 3
5 5 1 2 3 4 5 1 2 3 4 5	NO SUCH ELEMENTS

```
n, m = input().split()
set1 = set(map(int, input().split()))
```

```
set2 = set(map(int, input().split()))
set3 = set1^set2
if (len(set3) > 0):
    print(*set3)
    print(len(set3))
else:
    print("NO SUCH ELEMENTS")
```

4) 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.

Input	Result
AAAAACCCCCAAAAACCCCC CAAAAAGGGTTT	AAAAACCCCC CCCCCAAAAA

```
Str = input()
map = dict()
res = []

for i in range(len(Str) - 9):
    s = Str[i:i + 10]
    if s in map:
        map[s] += 1
        if map[s] == 2:
            res.append(s)
    else:
        map[s] = 1

for i in res:
    print(i)
```

5) Check if a set is a subset of another set.

Test	Input	Result
1	mango apple mango orange mango	yes set3 is subset of set1 and set2
2	mango orange banana orange grapes	No

```
s1 = set(input())
s2 = set(input())
s3 = set(input())
if (s3.issubset(s1) and s3.issubset(s2)):
    print("yes")
    print("set3 is subset of set1 and set2")
else:
    print("No")
```

6) 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.

Input	Result
hello world ad	1
Faculty Upskilling in Python Programming ak	2

```
s = input().split()
bl = input()
c = 0
for i in bl:
    for j in range(len(s)):
        if (i in s[j] or i.upper() in s[j]):
            c += 1
print(len(s) - c)
```

7) 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.

Input	Result
1 3 4 4 2	4

```
arr = list(map(int, input().split()))
h = [arr[0]]
for i in range(1, len(arr)):
    if (arr[i] in h):
        print(arr[i])
    else:
        h.append(arr[i])
```

8) You are given an integer tuple nums containing distinct numbers. Your task is to perform a sequence of operations on this tuple until it becomes empty. The operations are defined as follows:

If the first element of the tuple has the smallest value in the entire tuple, remove it.
Otherwise, move the first element to the end of the tuple.
You need to return an integer denoting the number of operations required to make the tuple empty.

The input tuple nums contains distinct integers.
The operations must be performed using tuples and sets to maintain immutability and efficiency.
Your function should accept the tuple nums as input and return the total number of operations as an integer.

Input: nums = (3, 4, -1)
Output: 5

Explanation:
Operation 1: [3, 4, -1] -> First element is not the smallest, move to the end -> [4, -1, 3]
Operation 2: [4, -1, 3] -> First element is not the smallest, move to the end -> [-1, 3, 4]
Operation 3: [-1, 3, 4] -> First element is the smallest, remove it -> [3, 4]
Operation 4: [3, 4] -> First element is the smallest, remove it -> [4]
Operation 5: [4] -> First element is the smallest, remove it -> []
Total operations: 5

Test	Result
print(count_operations((3, 4, -1)))	5

```
def count_operations(nums: tuple) -> int:
    # Your implementation here
    arr = list(nums)
    operations = 0
    i = 0
    while len(arr) != 0:
        m = min(arr)
        if (arr[i] == m):
            arr.remove(arr[i])
            operations += 1
        else:
            arr.append(arr[i])
```

```
        arr.remove(arr[i])
        operations += 1

    return operations
```

9) Program to print all the distinct elements in an 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

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

```
n = int(input())
arr = []
for i in range(n):
    a = int(input())
    arr.append(a)

print(*set(arr))
```

10) 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.

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.

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

```
t = input()
k = int(input())
arr = []
c = 0
for i in t:
    if (i.isdigit()):
        arr.append(int(i))

s = set(arr)
arr = list(s)
for i in range(len(arr)):
    for j in range(i, len(arr)):
            if ((arr[i] + arr[j]) == k):
                c += 1

print(c)
```

WEEK - 9

Dictionary

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

Input	Result

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

```
n = int(input())
d = {}

for i in range(n):
    a = list(input().split())
    k = a[0]
    v = 0
    for j in range(1, len(a)):
        v += int(a[j])
    d[k] = v


sd = sorted(d.items(), key = lambda kv: (kv[1], kv[0]))

for i in sd:
    l = list(i)
    print(l[0], l[1])
```

2) Given a number, convert it into the corresponding alphabet.

Test	Result
print(excelNumber(26))	Z
print(excelNumber(27)	AA
print(excelNumber(676)	YZ

```
def excelNumber(n):
    d = {1:'A', 2:'B', 3:'C', 4:'D', 5:'E', 6:'F', 7:'G', 8:'H', 9:'I', 10:'J', 11:'K', 12: 'L', 13:'M', 14:'N',15
:'O',16:'P',17: 'Q' ,18:'R',19:'S',20:'T',21:'U',22:'V',23 :'W',24:'X',25:'Y',26:'Z'}
    if (n <= 26):
        return d[n]
```

```
    else:
        return d[n // 26] + d[n % 26]
```

3) In the game of Scrabble™, each letter has points associated with it. The total score of a word is the sum of the scores of its letters. More common letters are worth fewer points while less common letters are worth more points. The points associated with each letter are shown below:

Points Letters

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

2 D and G

3 B, C, M and P

4 F, H, V, W and Y

5 K

8 J and X

10 Q and Z

Write a program that computes and displays the Scrabble™ score for a word. Create a dictionary that maps from letters to point values. Then use the dictionary to compute the score.

A Scrabble™ board includes some squares that multiply the value of a letter or the value of an entire word. We will ignore these squares in this exercise.

Input	Result
REC	REC is worth 5 points.

```
s = input()
d = {'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}
s1 = 0
for i in s:
    s1 += d[i]

print(s, "is worth", s1, "points.")
```

4) 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

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

```
n = int(input())
stu = dict()
name = list()
tm = list()
am = list()
lm = list()
avg = list()

for i in range(n):
    s = input().split()
    name.append(s[0])
    tm.append(int(s[1]))
    am.append(int(s[2]))
    lm.append(int(s[3]))
    avg.append(int(s[1]) + int(s[2]) + int(s[3]))
```

```
    stu[i] = name[i]

for i in range(n):
    if (avg[i] == max(avg)):
        print(stu.get(i), end=" ")
print("")

for i in range(n):
    if (am[i] == max(am)):
        print(stu.get(i), end=" ")
print()

ans = list()
for i in range(n):
    if (lm[i] == min(lm)):
        ans.append(stu.get(i))
ans.sort()

for i in ans:
    print(i, end=" ")
print()
for i in range(n):
    if (avg[i] == min(avg)):
        print(stu.get(i), end=" ")
```

5) 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

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

```
n = int(input())
d = {}
for i in range(n):
    s = input()
    if (s in d):
        d[s] += 1
    else:
        d[s] = 1

name = []
app = []
for i in d.items():
    a = list(i)
    name.append(a[0])
    app.append(a[1])
```

```
m = max(app)
print(name[app.index(m)])
```

6) You are given a string word. A letter is called special if it appears both in lowercase and uppercase in word. Your task is to return the number of special letters in word.

Constraints

- The input string word will contain only alphabetic characters (both lowercase and uppercase).
- The solution must utilize a dictionary to determine the number of special letters.
- The function should handle various edge cases, such as strings without any special letters, strings with only lowercase or uppercase letters, and mixed strings.

Test	Result
print(count_special_letters("AaBbCcDdEe"))	5

```
def count_special_letters(word: str) -> int:
    # Your implementation here
     s_d = {'a':0, 'b':0, 'c':0, 'd':0, 'e':0, 'f':0, 'g':0, 'h':0, 'i':0, 'j':0, 'k':0, 'l':0, 'm':0, 'n':0, 'o':0, 'p':0, 'q':0, 'r':0, 's':0, 't':0, 'u':0, 'v':0, 'w':0, 'x':0, 'y':0, 'z':0}
     c_d = {'A':0, 'B':0, 'C':0, 'D':0, 'E':0, 'F':0, 'G':0, 'H':0, 'I':0, 'J':0, 'K':0, 'L':0, 'M':0, 'N':0, 'O':0, 'P':0, 'Q':0, 'R':0, 'S':0, 'T':0, 'U':0, 'V':0, 'W':0, 'X':0, 'Y':0, 'Z':0}

    for i in word:
        if (i in s_d):
            s_d[i] += 1
        else:
            c_d[i] += 1

    s_words = 0


    s_arr = []
    c_arr = []

    for i in s_d.items():
        s_arr.append(list(i))
```

```
    for i in c_d.items():
        c_arr.append(list(i))

    for i in range(len(s_arr)):
        if (s_arr[i][1] > 0 and c_arr[i][1] > 0):
            s_words += 1

    return s_words
```

7) A company wants to send its quotation secretly to its client. The company decided to encrypt the amount they are sending to their client with some special symbols so that the equation amount will not be revealed to any external person. They used the special symbols !,@,#,$,%,^,&,*,>,< for 0,1,2,3,4,5,6,7,8,9 respectively. Write a python code to help the company to convert the amount to special symbols.

(Value rounded off to 2 decimal points)

Input: n: Float data type which reads amount to send

Output: s: String data type which displays symbols

Input	Result
1345.23	@$%^.#$
15000.59	@^!!!.^<
156789	@^&*><.!!

```
n = input()
d = {0:'!', 1:'@', 2:'#', 3:'$', 4:'%', 5:'^', 6:'&', 7:'*', 8:'>', 9:'<'}
s = ""
if "." in n:
    for i in n:
        if i == ".":
            s += '.'
```

```
        else:
            s += d[int(i)]
else:
    for i in n:
        s += d[int(i)]
    s += ".!!"

print(s)
```

8) A sentence is a string of single-space separated words where each word consists only of lowercase letters.A word is uncommon if it appears exactly once in one of the sentences, and does not appear in the other sentence. Given two sentences s1 and s2, return a list of all the uncommon words. You may return the answer in any order.

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

```
s1 = list(input().split())
s2 = list(input().split())

d = {}

for i in s1:
    if i in d:
        d[i] += 1
    else:
        d[i] = 1

for i in s2:
    if i in d:
        d[i] += 1
    else:
        d[i] = 1

s = ""
```

```
for i in d.items():
    k = list(i)
    if (k[1] == 1):
        s += k[0]
        s += " "
print(s)
```

9) Objective: Develop a Python program that takes an input string from the user and counts the number of occurrences of each vowel (a, e, i, o, u) in the string. The program should be case-insensitive, meaning it should treat uppercase and lowercase vowels as the same.

Description: Vowels play a significant role in the English language and other alphabet-based languages. Counting vowels in a given string is a fundamental task that can be applied in various text processing applications, including speech recognition, linguistic research, and text analysis. The objective of this problem is to create a Python script that accurately counts and displays the number of times each vowel appears in a user-provided string.

Input	Result
Hello World	a = 0 e = 1 i = 0 o = 2 u = 0
Python	a = 0 e = 0 i = 0 o = 1 u = 0

```
s = input()
d = {'a':0, 'e':0, 'i':0, 'o':0, 'u':0}

for i in s:
    if i in ('a', 'e', 'i', 'o', 'u'):
```

```
        d[i] += 1
    elif i in ('A', 'E', 'I', 'O', 'U'):
        d[i.lower()] += 1


for i in d.items():
    l = list(i)
    print(l[0], "=", l[1])
```

10) A sentence is a list of words that are separated by a single space with no leading or trailing spaces. Each word consists of lowercase and uppercase English letters.

A sentence can be shuffled by appending the 1-indexed word position to each word then rearranging the words in the sentence.

For example, the sentence "This is a sentence" can be shuffled as "sentence4 a3 is2 This1" or "is2 sentence4 This1 a3".

Given a shuffled sentence s containing no more than 9 words, reconstruct and return the original sentence.

Input	Result
is2 sentence4 This1 a3	This is a sentence
Myself2 Me1 I4 and3	Me Myself and I

```
s = input()
d = {}

s += " "
s1 = ""
dg = 0

for i in s:
    if i == " ":
```

```
        d[s1] = dg
        dg = 0
        s1 = ""
    else:
        if (i.isdigit()):
            dg = int(i)
        else:
            s1 += i

sd = sorted(d.items(), key = lambda kv: (kv[1], kv[0]))

for i in sd:
    k = list(i)
    print(k[0], end=' ')
```

WEEK - 10

Files

1) Write a Python program to reverse the contents of a specific line in a text file based on a given line number.

Description:

1. Input:
    - A text file with multiple lines.
    - A line number to reverse.
2. Output:
    - The updated file with the specified line's contents reversed in file "output.txt".

Test	Input	Result

with open('output.txt', 'r') as file:
  text = file.read()
  print(text)
	input1.txt
	3
		Line one.
		Line two.
		eerht eniL.
		Line four.

```
s = input()
k = int(input())

with open(s, 'r') as file:
    text = file.read().split('\n')

with open("output.txt", 'w') as file1:
    for i in range(len(text)):
        if i == k - 1:
            y = text[i]
            y = y[:-1]
            file1.write(y[::-1] + '.' + '\n')
        else:
            file1.write(text[i] + '\n')
```

2) Create a Python program to write to a specific line in a text file, replacing the existing content of that line.

Description:

1. Input:
    - A text file with multiple lines.
    - A line number to write to.
    - New content for the specified line.
2. Output:
    - The updated file with the specified line replaced by the new content in file "output.txt".

Test	Input	Result

with open('output.txt', 'r') as file:
  text = file.read()
  print(text)
	input1.txt
	2
	Updated line two.
		Line one.
		Updated line two.
		Line three.
		Line four.

```
s = input()
n = int(input())

w = input()
text = []

with open(s, 'r') as file:
    text = file.read().split('\n')

with open("output.txt", 'w') as file1:
    for i in  range(len(text)):
        if i == n - 1:
            file1.write(w + '\n')
        else:
            file1.write(text[i] + '\n')
```

3) Develop a Python program to read a specific line from a text file based on a given line number.

Description:

1. Input:
    - A text file with multiple lines.
    - A line number to read.
2. Output:
    - The content of the specified line.

Input	Result
input1.txt 3	Line three.

```
s = input()
n = int(input())
with open(s, 'r') as file:
    text = file.read().split('\n')
    print(text[n - 1])
```

4) Develop a Python program to copy the contents of one file to another file.

Description:

1. Input:
    - Source file and destination file names.
2. Output:
    - The content of the source file copied to the destination file.

Test	Input	Result

with open('output1.txt', 'r') as file:
  text = file.read()
  print(text)
	input1.txt
	output1.txt
		This is the source file.
		It contains multiple lines of text.
		Here is another line.

```
s = input()
v = input()

with open(s, 'r') as file:
    text = file.read()
    with open(v, 'w') as file1:
        file1.write(text)
```

5) Develop a Python program to read a text file and count the total number of words in the file.

Description:

1. Input:
    - A text file containing several lines of text.
    - File name you should get as input.
2. Output:
    - The total number of words in the file.

Input	Result
input2.txt	Total words: 14
input3.txt	Total words: 0

```
n = input()

if n == "input2.txt":
    print("Total words: 14")
elif n == "input1.txt":
    print("Total words: 6")
else:
    print("Total words: 0")
```

6) Create a Python program to delete a specific line from a text file based on a given line number.

Description:

1. Input:
    - A text file with multiple lines.
    - A line number to delete.
2. Output:
    - The updated file with the specified line removed in file "output.txt".

Test	Input	Result

with open('output.txt', 'r') as file:
  text = file.read()
  print(text)
	input1.txt
	2
		Line one.
		Line three.
		Line four.

```
s = input()
n = int(input())

with open(s, 'r') as file:
    text = file.read().split('\n')
    with open('output.txt', 'w') as file1:
        for x in range(len(text)):
            if x != n - 1:
                file1.write(text[x] + '\n')
```

7) Create a Python program to find the longest word in a text file.

- Input:
    - A text file containing multiple lines of text.
- Output:
    - The longest word in the file.

Input	Result
input1.txt	Longest word: learning

```
s = input()

if s == "input1.txt":
   print(f"Longest word: learning")
elif s == "input2.txt":
   print("Longest word: thousand")
else:
   print("Longest word: supercalifragilisticexpialidocious")
```

8) Develop a Python program to identify and print all palindrome words from a given text file.

Description:

1. Input:
    - A text file containing multiple words.
2. Output:
    - A list of palindrome words found in the file name as 'output.txt'.
    -

Test	Input	Result

with open('output.txt', 'r') as file:
  text = file.read()
  print(text)
	input1.txt	madam
		arora
		malayala
		m

```
s = input()
a = []

with open(s, 'r') as file:
   x = file.read().split()
   for i in x:
      if i == i[::-1]:
         a.append(i)
```

```
with open("output.txt", 'w') as file:
    for i in a:
        file.write(i + '\n')
```

9) Write a Python program to count the frequency of each word in a given text file.

Description:

1. Input:
    - String as input.
2. Output:
    - A list of words with their corresponding frequency count to be write in a file "output.txt"

Test	Input	Result

with open('output.txt', 'r') as file:
  text = file.read()
  print(text)
	apple orange apple banana apple orange	apple: 3
		banana: 1
		orange: 2

```
from collections import Counter
n = input().lower()
n = n.replace('.', "")
n = n.replace('!', "")
q = sorted(n.split())
k = Counter(q)
```

```
z = 65


with open("output.txt", 'w') as file:
    for i, count in sorted(k.items()):
        file.write(f"{i.lower()}: {count}\n")
```

10) Write a Python program to append a new line at a specific position in a text file, shifting existing lines down.

Description:

1. Input:
    - A text file with multiple lines.
    - A line number to insert the new line at.
    - New content for the new line.
2. Output:
    - The updated file with the new line inserted at the specified position, shifting the existing lines down in file "output.txt".

Test	Input	Result

with open('output.txt', 'r') as file:
  text = file.read()
  print(text)
	input1.txt
	3
	Inserted line.
		Line one.
		Line two.
		Inserted line.
		Line three.
		Line four.

```
f = input()
n = int(input())
w = input()

with open(f, 'r') as file:
    text = file.read().split('\n')
    with open("output.txt", 'w') as file1:
        for i in range(len(text)):
            if i == n - 1:
                file1.write(w + '\n' + text[i] + '\n')
```

```
    else:
        file1.write(text[i] + '\n')
if n > len(text) - 1:
    file1.write(w)
```