INTRODUCTION TO PROGRAMMING

```
"""
Set of inst. that tell computer how to perform the task
computer only knows 0 and 1
0 and 1 is machine code , low level language
to communicate with computer we use source code (high level
language(c++,java,python))---->(by compiler) machine code
high level language nearer to human language
python--->byte code--->interpreter--->high level language
"""

#INTRODUCTION TO THE PYTHON
"""
One of the most famous and preferred language
because
  very easy to read and understand
  beginner friendly
  conceited code(kisi aur ka code as a module or package use ho jayega)
  open source
  versatile(webdev, ML,AI ,IOT etc)
"""

#FIRST CODE
print("Hello world!")

#BASICS PROGRAM IN PYTHON
"""
print hello X
print"hello" X
print(hellow) X

have the proper syntax

print("hellow") CORRECT
"""

#NEW LINE
print("hello\nworld")

"""
\n k bada ka next line
"""

#COMMENT
"""
We do not want that the line will execute in the code just for add the information to the code
single line comment (#)
multiline comment triple quotes
```

```
"""

#PYTHON INDENTATION
"""
print("")  CORRECT
    print("") X

if cond 1:
print("a")   X

if cond 1:
  print("a")  CORRECT
"""

#PYTHON CLI
"""
Command line interface
REPL-Read evaluate print loop
nechay on the terminal direct operation
'''PS C:\Users\HP\OneDrive\Documents\Desktop\CODE\python\NEW> 2+4
6'''

"""
```

list of operators

```
"""
Arithmetic operators
Assignment operators
Comparison operators
logical  operators
identity operators
membership operators
bitwise operators
"""

#Arithmetic operators
"""
+ = a+b
- = a-b
* = a*b
/ = a/b
% = a%b   (return the reminder)
** = (a)^power
// = floor division  (return nearest whole number)
"""
print(4+3)
print(4-3)
print(3*3)
print(4/4)
print(5%2)
print(2**2)
print(4//3)

#Assignment operators
"""
=  (n1=1 ; n2=n1)
+=  n2+=n1(n2=n2+n1)
-=  n2-=n1(n2=n2-n1)
=  n2=n1(n2=n2*n1)
/=  n2/=n1(n2=n2/n1)
%=  n2%=n1(n2=n2%n1)
//= n2//=n1(n2=n2//n1)
*= n2=n1(n2=n2*n1)
&=
|=
^=
>>=
<<=
"""

#comparison operators
"""
== equal
!= not equal
> greater than
< less than
>= greater than or equal to
<= less than or equal to


inka output always bollen hoga ( True(1) or False(0))
"""

print(4==3)
print(2<5)
print(10>8)
print(8!=7)

#logical operator
"""
and - return true if both are true
or - return true if one of the are true
Not - reverse the result if true then false

"""
```

```
print(2==2 and 1==1)
print(2==2 or 2==0)
print(not 2==2)


#IMPORTANT

#Membership operator
"""
in -- return True if a sequence with the specified value is present in the object
not in -- return True if a sequence with the specified value is not present in the object

useful in set data , list data , tuple data

"""

fruit=["apple","banana","cheery"]
print("if banana is present in fruits:","banana" in fruit)
print("if mango is present in fruits:","mango" is not  fruit)

#Bitwise Operators
"""
& - ANS
| - OR
^ - XOR
~  - NOT
<<  - ZERO FILL LEFT SHIFT
>> - SIGNED RIGHT SHIFT
"""
a=5
b=3
print(a&b)
print(a|b)
print(a^b)



#question solving
#calculate of the sphere

radius=int(input("Enter the radius of the radius:"))
area=float((4/3)3.14*radius*3)
print("The volume of the sphere is :",area)


#operator precedence
"""
BODMAS
```

```
() bracket
** exponacatio
/ , // , %
*
+ ,-
bitwise shift ->>,<<
rest of the bit wise & | ^
comparison operator - == , != , > ,<
logical (not , and , or )
"""
#eg - 3+2**4/2*5-8//2   ans - 39


#Sone in built function
"""
used to find data type of the variable
"""

psit=89
print(type(psit))

#Type casting
"""
converting one data type into the other data type

"""
che=float(7)
print(type(che))

l=2
m=8
n=9
str_l=str(l)
str_m=str(m)
str_n=str(n)

final_str=str_l+str_m+str_n
print(final_str)
print(type(final_str))

#practice question
# WAP to convert the temperature from the celusisi to the kelvin

temp=float(input("Enter the temperature in the celuse:"))
convert=temp+273
print("The given temperature in the kelvin is:",convert)
print(type(convert))
```

VARIABLES

```
"""
it is the place where the data is stored(memory ka name then usmai value)

data can be of different type
int , string , float ,boolean (True,False) ,none
'''variable do not need to be declared with any particular type in python '''
dynamically typed(kuda pata hota hai usko)

name of the variable must be descriptive
"""

#RULES
"""
Start with the letter or the underscore letter
Variable cannot be started with number
Only contains the letter (A-Z,a-z) and number (0-9) and underscore_
Case sensitive A and a have the different value
We can not use the python keyword
No special character is required except underscore(_)

"""
#STRING
name='isha'

#INT
roll_no=50

#PERGENTAGE
per=99

#BOOLEAN
is_student=True

#PRINTING THE VARIABLE
print(name, roll_no,per,is_student)

#UPDATION IN SAME VARIABLE
per=98.990
print(name, roll_no,per,is_student)

#TYPE OF THE VARIABLE
print(type(name))
print(type(roll_no))
print(type(per))
print(type(is_student))
```

```
#CHECK THE ID OF THE VARIABLE
print(id(name))
print(id(roll_no))
print(id(per))
print(id(is_student))

#PRINTING WITH THE DESIRED LINE
print("My name is:",name)
print("My roll no is:",roll_no)
print("My percentage is:",per)
print("I am a student:",is_student)


#PRINT STATEMENT IN THE SAME LINE
print("My name is:",name ,"My roll no is:",roll_no,"My percentage is:",per,"I am a
student:",is_student)


#we can concatenate the same data type only
"""
string+string

"""
print("my name is:"+name)

#PRINT EXPRESSION
print("My percentage has changed to",per-1)

#PRINT WITH SEPARATOR
print(name,roll_no,per,is_student,sep="->")
```

Numeric data type

```
"""
 Numeric data type

1.Int -> 100,-9
2.Float ->2.4,-1.5
3.Complex -> 1+2j,6+250j
"""

#Text data Type
"""
1.string ->sequence of char in closed by ' ' or " " ->"gagan"
"""

#boolean data type
```

```
"""
True or False
"""

#Sequence data typeṣ
"""
1.List - sequence of data (eg- sequence of integer [1,2,3], string ['a','b'])   unique and modifiable
    can store different type of the data               my_list = [1, "hello", 3.14, True]
                                              Mutable
2.Tuple - present in the () brackets and duplicate value are also allowed  (eg-(1,2,3,4,5,6,7)) not
modifiable
    can store different type of the data               my_tuple = (1, "Hello", 3.14, [1, 2, 3])
                                              immutable
"""

#Mapping data type
# ->  # Dictionary Data type
"""
Store key value pairs
   eg - student1 = {
    "name": "xyz",
    "rollno": 90
}

            Mutable

"""
# -> #Set data type
"""
unordered collection of unique items
eg-fruit={"apple","banana","grappies"}

            Mutable

my_set = {1, 2.5, "hello", True}
print(my_set)
"""

#None data type
"""
Store nothing
variable jis k pass koi value nahi hai
"""

#ASCII and Unicode value
"""
```

ASCII-American Standard for information interchange

use to represent the char as the numeric code

```
    A-Z--65-90
      A=65 and so on
    a-z -- 97-122
      a=97 and so on
    0-9 -- 48-57
      '0'=48
    Space -- 32
"""

#Inbuilt function (ord())
"""
It will tell the direct the ascii value
word on the string length only one
"""
char="a"
print(ord(char))

# chr()
"""
It will tell the character by the number

"""
a=97
print(chr(a))
```

#Control statement

```
"""
These statements allow us to control the flow of the program .
"""
#Conditionals Statement
"""
if,if-else
nested
Else if ladder
ternary
switch
"""

#IF-ELSE
"""
if raining==True
  print("take umbrella")
else
  print("Do not take the umbrella")


syntax--
```

```
if condition:
 // statement 1//
else :
 //statement 2//

indation is most important  else the code will not work
"""

#Check the number is positive or not
num=int(input("Enter the number:"))
if num>=0:
    print("Positive Number")
else :
    print("Negative Number")

#Check the number is even or odd
num=int(input("Enter the number:"))
if num%2==0:
    print("Even Number")
else :
    print("Odd Number")

#Check Profit or loss
cost_price=float(input("Enter the cost price of the time:"))
sell_price=float(input("Enter the sell price of the time:"))

profit=sell_price-cost_price
loss=cost_price-sell_price
if sell_price > cost_price:
    print("The seller make the profit:",profit,"Rs")
elif sell_price==cost_price:
    print("No profit , No lose")
else:
    print("The seller is made the  loss:",loss,"Rs")


#Percentage of the student
per=float(input("Enter the percentage the of the student:"))
if per<=100 or per<=81:
    print("Very good")
elif per<=80 or per<=61:
    print("Good")
elif per<=60 or per<=41:
    print("Average")
elif per<=40:
    print("Fail")
else:
    print("Invalid Percentage")
```

```
#Multiple Condition using "and" and "or"
"""
and - jab dono condition jin mai comparison hoga tb condition calygi
or - jab dono condition mai ek sahi hogi tb cal jyegga
implemented in the above question

"""


#Check the number is four digit or not
number=int(input("Enter the number:"))
if number>999 and number<=9999:
   print("Enter the number is the three digit number:")
else :
   print("The number is no three digit number")

#Check greatest among the three numbers
num1=int(input("Enter the number one:"))
num2=int(input("Enter the number two:"))
num3=int(input("Enter the number three:"))

if num1>num2 and num1>num3:
   print("Num1 is the greater",num1)
elif num2>num1 and num2>num3:
   print("Num2 is the greater",num2)
elif num3>num1 and num3>num2:
   print("Num3 id the greater",num3)

#Nested If - Else
"""When we have to take the decision between the multiple conditions"""
num1=int(input("Enter the number one:"))
num2=int(input("Enter the number two:"))
num3=int(input("Enter the number three:"))

if num1>num2:
   if num1>num3:
      print("num1 is the greatest number:",num1)
   else :
      print("num3 id the greatest number:",num3)
else:
   if num2>num3:
      print("num2 is the greatest number:",num2)
   else :
      print("num3 is the greatest number:",num3)

#Take positive integer input and tell if it is divisible by 5 or 3 but not divisible by 15

d=int(input("Enter the Number:"))
```

```
if d%5==0 or d%3==0:
    if d%15!=0:
        print("The digit is divisible by the 5 or 3 but not by 15")
    else:
         print("The digit is divisible by the 5 or 3 and 15 too")


#Match case (python 3.10)
"""
ek valse is different values se mat karna hai then we use it
in c switch case

syntax:
match x:
case 1:st1
case 2:st2
case 3:default
"""

#calculator
c1=int(input("Enter the number one:"))
c2=int(input("Enter the number two:"))

sing=input("Enter the operation:")

match sing:
    case"+":print("sum:",c1+c2)
    case"-":print("diff:",c1-c2)
    case"*":print("mul:",c1*c2)
    case"/":float(print("divide:",c1/c2))
    case"_":print("Enter the valid operator")


#Ternary Operation
"""only used for two comparison between the two condition only : sedha haa ya na bata hai
work done within one line
"""
#eg-
rain=1
umb="yes" if rain==True else "no"
print(umb)

#Check the entered number is prime or not using ternary statement
n1=int(input("Enter the number:"))
ev="even" if n1%2==0  else "odd"
print(ev)
```

Input Keyword

```
"""
name=input("Enter your name")

"""


name=input("Enter you name:")
print(name)

#Input is always is taken as the string
"""
for overcoming this we use TYPE CASTING
  TYPE CASTING= Convert on data type into other
    rollno=int(input("Enter the roll no))
"""

age=input("Enter the age:")
print(type(age))   # it will give string so we will use the type casting

agee=int(input("Enter the age:"))
print(type(agee))


#Sum of 2 given number
num1=int(input("ENter the number one:"))
num2=int(input("Enter the number two:"))

sum=num1+num2

print("The sum of given number is",sum)
```

loop

```
"""when we have  to do any task repeatedly
"""
#There are two type of the loops
"""
for
while
"""

#for
"""
for i in range (1,10):
#code

initialization , kaha se start hoga , kha katama hoga (start, stop)
start - inclusive
```

```
stop - exclusive

range function will have the value by default increase by the one
unless we mention the step (1,10,2)

range(start, end , step)
start - kha se start hoga
end - kha per khatma hoga
step- itna increment karna hai

if we do not give the starting point then it will take it as the zero
"""
#given the start and end
for i in range (1,11):
    print(i,"PSIT")
#given the step
for i in range (1,11,2):
    print(i,"PSIT")
#no starting point
for i in range (11):
    print(i,"PSIT")

# when we have to print only then we have just skip the i
for _ in range(10):
    print("PSIT")

#print the element of the list using the loop
list1=[1,20,3,4,5,6,7,8]
for i in list1 :
    print(i)

# While loop
"""
runs till condition is true
befour every iteration is true

i=0
while i<10:
code
i+=1
"""

i=2
while i<100:
    print(i)
    i+=1
```

pattern printing

```
"""
we have to work on three things
Rows =n
columns =m
what to print =*
"""
n=int(input("Enter the number of the rows:"))
for i in range(n):
    print("*"*5)


n=int(input("Enter the number of the rows:"))
for _ in range(n):
    for i in range(1,n+1):
     print(i,end="")
    print("")


n=int(input("Enter the number of the rows:"))
for i in range(1,n+1):
    for j in range (1,i+1):
     print(j,end="")
    print("")

n=int(input("Enter the number of the rows:"))
for i in range(n,n+1):
    for j in range (65,i+1):
     print(chr(j),end="")
    print("")

n=int(input("Enter the number of the rows:"))
for i in range(1,n):
   print(" "*n-i, end="")
   for j in range (1,2*i):
      print(j,end="")
   print()
```

Function

```
"""
what and why
type of function
creating a function
calling a function
arguments
types of arguments
```

```
difference bw parameters and arguments
return type
nested function
pass by value
pass by reference
built in function
"""

#what and why
#sum of all no till then
n=int(input("Enter the number:"))
sum=0
for i in range(0,n+1):
    sum=sum+i
print(sum)

"""function are block of the reusable code that perform a specific task"""
"""input------> function------> output"""


#type of function
"""built in function - print, sum etc"""
"""use defined function - defined by us as per need """


#creating a function
""" def function_name(parameters):"""
"""        #statement              """ #body
"""       return repression       """
#         function return ^

# sum of the two numbers
def sum(n1,n2):
    ans=n1+n2
    return ans

#Calling a function
"""function_name(argument1,argument2)"""
print("The sum of the two number is :",sum(1,1))
#                              calling function^


#Write a function to print the hello world
def prithello():
    print("hello world!")


prithello()
```

```
#Tpe of the arguments
"""
default argument
keyword arguments(named argument)
position argument
arbitrary arguments (variable-length arguments *args and **kwargs)
#**kwargs
"""

def add(n1,n2):
    sum=n1+n2
    return sum
#positional argument
print("The add is",add(1,2))

#keyword arguments(named argument)
print("The add is",add(n1=1,n2=2))

def add(n1=0,n2=0):#they already have the default values
    sum=n1+n2
    return sum
#default argument
print("The add is",add(1))
"""n2=0 by default """

#arbitrary arguments (variable-length arguments *args **kwargs)
"""
def addall number(*args)
 #args will store in the form of the tuple
"""
def addall number(*args):
    sum=0
    for i in args:
        sum+=i
    return sum

print("the sum is",addall number(1,2,3,4,5))

#**kwargs
"""key value pairs arguments"""
"""
def student_info(**kwargs):
keywords- as a dict they will pass

"""
def student_info(**kwargs):
    for x,y in kwargs.items():
        print(x,"is",y)

student_info(name="gagan",age=19)
student_info(name="ritika",age=19)

#write a program using function to print the sum off all the number from 1 to n
def sum (n):
    sum=0
    for i in range(0,n+1):
        sum+=i
    return sum

n=int(input("Enter the value of the n:"))
print("the sum of the number 1 to",n,"is:",sum(n))

#function will define befour the calling it

#Nested function
"""function in another function is called the nested function"""
def out_function():
    x=1 #variable in the outer function

    def inner_function():
        y=2 #variable in the inner function
        result= x+y
        return result

    return inner_function()

output=out_function()
print("The value of the inner and the outer function will be:",output)

#Pass by value and Pass by the reference
"""
-pass by value (immutable object -string ,int ,float,tuples)
-when passed to function a copy of the object is created and assigned to local
variable in the function
-any change made to local variable in the function then function, do not affect the original
variable outside the function
"""

def addOne(x):
    x=x+1
    print("Inside function:",x)

x=5
addOne(x)
print("The value of the x outside of the function is :",x)
```

```
#pass by the reference
"""
used for the mutable objects-list ,dict
a reference is o actual is passed to the function
change in the object in the function will as showcase outside of the function it self
"""
#pass by the reference
define modify list(li):
    li.append(4)
    print("Inside list the value of the list is :",li)

li=[1,2,3]
modifylist(li)
print("The value of the list outside of the function is:",li)

define modify list(li):
    lis=[6,7,8] # new object so the value will not modify
    print("Inside list the value of the list is :",lis)

li=[1,2,3]
modifylist(li)
print("The value of the list outside of the function is:",li)

#built in function in python
"""
print
sum
min
max
etc....
"""

#Question
"""WAP to print the factorial of the number using function"""

def factorial (n):
       if n==1 or n==0:
        return 1
       else:
         fact=1
         for i in range (1,n+1):
          fact*=i
       return fact

n=2
print(factorial(n))
```

key characteristics of recursive function

```
"""
factorial of n
   n!=n[(n-1)(n-2)(n-3).....*1]
   (n-1)!=(n-1)(n-2)(n-3).....*1
   n!=n*(n-1)!
"""
"""factrial(n)
   factorial(n-1)*n
"""

#What is recursion?
"""
def recurse():
   recurse() -->recursive call
recurse()  -->recursive call

it call it self to solve big from small sub problem
"""

#factorial without the recursion
def fact(n):
    fact=1
    for i in range(1,n+1):
        fact*=i

    return fact
print(fact(5))

#factorial with the recursion
def fact(n):
    if n==1:
     return 1
    f=n*fact(n-1)
    return f
print(fact(5))

#base code and recursive case
def fact(n):
    if n==1:#base case
     return 1#base case
    f=n*fact(n-1)#recursive call
    return f

#The call stack and Recursive calls

#Practice question
```

```
#factorial n
def fact(n):
    if n==0:
     return 1
    f=n*fact(n-1)
    return f
n=int(input("Enter the number n:"))
print(fact(6))

#print the from n to 1
define noone(n):
    if n==0:
        return
    print(n)
    ntoone(n-1)
ntoone(5)

#sum till the n
def sum(n):
    if n==1:
        return 1
    s=n+sum(n-1)
    return s

print(sum(3))

#calculate the power using recursion
def power(a,b):
    if b==0:
        return 1
    p=a*power(a,b-1)
    return p

print(power(2,2))

#fibonacci series
def fib(n):
    if n==1:
        return 0
    elif n==2:
        return 1
    else:

      f=(fib(n-1)+fib(n-2))
      return f
n=5
for i in range(1,n+1):
  print(fib(i))
```

List

```
#python collections (arrays)
"""
Lists
Tuples
Set
Dictionary

these data help us to store the collection of the data in the one variable
"""

#list
"""
Allow to store the multiple items
list=["a","b"]
the items have the index starting from the zero
item are ordered (perest in same order)
mutable - updation is allowed
duplicates are allowed
also store items with the different data types
"""
fruits=["apple","mangoes","cherry"] # create a list
print(fruits) #print the list
print(type(fruits)) #check the type of the list
print(len(fruits)) # check the length of the list

#check an item is in list  or not
if "banana" is fruits:
   print("It is in list")
else:
   print("Not present in the list")

if "banana" not in fruits:
   print("It is not in list")
else:
   print("present in the list")

#accessing items of a list
"""
Indexing  -> 0 , 1 ,2 ,n-1
negative indexing ->opp last is -1 and so on
range of indexes ->list[starting:ending] (inclusive:exclusive)
range of negative indexes
"""
#fruits=["apple","mangoes","cherry"]
```

```
print(fruits[1])
print(fruits[-2])
print(fruits[0:3])#sublist
print(fruits[0:4])#print whole list using positive indexing
# we cannot print the whole list using the negative indexing

#Adding element to a list
"""
append() - add item to the end of the list  (list.append(n))
insert() - want to insert the element the specific position  (list.insert(position,n))
extend() -when we merge the two list (list.extend(list2))
"""

num=[1,2,3,4,5]
num.append(6)
print(num)

num.insert(0,0)
print(num)

nuum=[7,8]
num.extend(nuum)
print(num)

#Removing elements from a list
"""
remove()-it removes the specified time . (list.remove(n))
pop()-it remove item of the target index , else last time (list.pop())
"""
numm=[1,2,3,4,5,6]
numm.remove(1)
print(numm)
numm=[1,2,3,4,5,6]
numm.pop(0)
print(numm)
numm=[1,2,3,4,5,6]
numm.pop()
print(numm)


#changing time in a list
"""
At an index- list[n]=n
In the range- list[n:m]=[n,m]
"""

li=[1,2,3,4]
```

```
li[3]=8
print(li)

li=[1,2,3,4]
li[0:2]=[3,4]
print(li)

#shorting a list
"""
ascending=small to big   (list.sort())  by default is happens
descending=big to small   (list.sort(reverse=True))
"""

op=[56,1,67,9,166]
op.sort()
print(op)

op=[56,1,67,9,166]
op.sort(reverse=True)
print(op)

#reverse function
"""it will reverse the list first element is last and last is first"""
op=[56,1,67,9,166]
op.reverse
print(op)

#list comprehension
"""when we want to make a new list from time of existing list"""

#we want a list of item greater than 22  (using loop)
l1=[21,22,23,24,25]
newlist=[]
for i in l1:
    if i>22:
        newlist.append(i)
print(newlist)

#using list comprehension want a list of item greater than 22
fruits=["apple","mangoes","cherry"]
newlist=[fruits for fruits in fruits if "a" in fruits]
print(newlist)

#copy the list in other list
a=new list.copy()
print(a)
```

```
#we can list add two list
lst1=[1,2,3,4,5,6,7]
lst2=[8,9,10,11,12,13]
print(lst1+lst2)


#nested list
nlist=[10,20,[30,40],50,60]
print(nlist[0])
print(nlist[1])
print(nlist[2])
print(nlist[2][0])
print(nlist[2][1])
print(nlist[3])
print(nlist[4])

#Question
qlist=[23,65,19,90]
qlist[-4]=19
qlist[-2]=23
print(qlist)

#or
qlist=[23,65,19,90]
qlist[0]=19
qlist[2]=23
print(qlist)


#question
qq=[1,2,3,4,5]
qq[1]=5
qq[4]=2
print(qq)

#or
qq=[1,2,3,4,5]
temp=qq[0]
qq[1]=qq[4]
qq[4]=temp
print(qq)
```

Tuples

```
"""Used to store multiple items in a variables"""
"""use round bracket tuple()"""
#Property
```

```
"""
Ordered
Immutable -operation not allowed
Duplicates allowed
Any data types
Mixed type of the data
"""

colours=("red","yellow","blue")

#single element in the tuple (use of the , after element)
fruite=("apple",)
#or
fruit=tuple("apple")

#checking the type of the tuple
print(type(colours))
print(type(fruite))
print(type(ffruite))

#Length of tuple
print(len(colours))
print(len(fruite))

#accessing items in tuple
"""positive indexing"""
print(colours[0])
print(colours[1])
print(colours[2])

"""NEgative indexing"""
print(colours[-1])
print(colours[-2])
print(colours[-3])

"""Range indexing positive"""
print(colours[0:3])

"""Range indexing negative"""
print(colours[-3:-1])
print(colours[-3:])#if we just tell starting point and no enf jitna tuple hai

#Want to check that there is time is in the list or not
if "blue" in colours:
   print("Green is there")
if "orange" not in colours:
   print("orange is not there!")
```

```
#Triverse a the tuble
for i in colours:
    print(i)


#concatenate the tuple
new_colurs=("magenta","creame")
print(colours+new_colurs)

#unpacking a tuple
"""jyse ki ek tuble mai kafi sare values hoti hai unko hata k alga alga variable mai dalana"""
colour1,colour2,colour3=colours
print(colour1,colour2,colour3,sep="->")

#Tuple VS Lists
"""
Iterating through a 'tuple' is faster than in a 'list'
'list' are mutable whereas 'tuples' are immutable
Tuples that contain immutable elements can be used as a key for a dis

"""

#reverse()
"""it iterate through a sequence through a sequence in reverse order"""
tp=('z','a','d','f','g','e','e','k')
for i in reversed(tp):
    print(i,end="",sep=",")

#or
tp=('z','a','d','f','g','e','e','k')
li=[]
for i in reversed(tp):
     li.append(i)
print(tuple(li))


tp=(10,11,12,13,14,15)
li=[]
for i in reversed(tp):
    li.append(i)
print(tuple(li))
```

Sets

```
"""container for storing multiple values in a single variable"""
"""set={"A","b"}"""
```

```
#Property of the sets
"""
Unordered  - item ka sequence nahi hota hai print in any order
Immutable  - update existing allowed , but can remove , add
Unindexed  -
Duplicates not allowed -*{"a","b","a"} X
Any datatype
Mix of different data type  -set={1,false,1.3,"no"}
"""
sett={"gagan","ritika","dhruv"}
print(sett)

#check length of the set
print(len(sett))

#check data type in the python
print(type(sett))

#accessing items of set
for x in sett:
    print(x)

#check if an item exists in a set
if "gagan" in sett:
    print("Gagan is there inn the set")
if "abhishek" not in sett:
    print("Abhishek is not there in the set")

#add element in the set (add())
sett.add("pagal")
print(sett)
"""do not add the same element again"""

#add another sequence in the set (update())
tp=("yuvraj","shyam")
sett.update(tp)
print(sett)

#remove element from the set (remove())
sett.remove("pagal")
print(sett)

# if we do not known that the value is there in the set or not and want to remove it
"""discard() if we use remove then if value is not there then , it will throu8gh the error"""
sett.discard("arayan")
print(sett)

#want to join the two sets
```

```
s1={'a','b','c'}
s2={'d','e','f'}
#print(s1+s2) -> not allowed
s3=s1.union(s2)
print(s3)

#or
s1.update(s2)
print(s1)

#keep only duplicate while joining
s={1,2,3,4}
p={3,4,5,6}
s.intersection_update(p)
print(s)

#keep all the value except duplicate
s={1,2,3,4}
p={3,4,5,6}
s.symmetric_difference_update(p)
print(s)

#Max and Min in the set
s1={1,2,3,4,5,6,7,8}
a=max(s1)
b=min(s1)

#Question 1
"""
Given three arrays , we have to find common elements in three sorted lists using sets
"""
l1=[1,5,10,20,40,80]
l2=[6,7,20,80,100]
l3=[3,4,15,20,30,70,80,120]

s1=set(l1)
s2=set(l2)
s3=set(l3)

s1s2=s1.intersection(s2)
fs=s1s2.intersection(s3)
fl=list(fs)
print(fl)

#Question 2
"""
Given three arrays , we have to find common elements in three sorted lists using sets
"""
```

```
l1=[1,5,5]
l2=[3,4,5,5,10]
l3=[5,5,10,20]

s1=set(l1)
s2=set(l2)
s3=set(l3)

s1s2=s1.intersection(s2)
fs=s1s2.intersection(s3)

lf=list(fs)
print(lf)


#Dictionary
"""we will store  the key value pairs"""
"""
phone dictionary
gagan- 9978575676
ritika- 8585794649

english dictionary
help-_______
gratitude-_________

key value pair
"""
#Creating a dictionary
"""
numbers={
"gagan":65432 , (key:value)
"ritika":76543 ,   (key:value)
"joy":54343      (key:value)
}
"""
```

Dictionary items

```
"""
Ordered - print in same order as store
changeable - updation is allow
unindexed - number se access nahi hogi
Duplicates not allowed - same key are not allowed
any data types - mixed
"""
#creating a Dictionary
phone={
   "gagan":345678,
    "ritika":87654,
    "dhruv":76543,
}

#printing a dictionary
print(phone)

#Checking the type
print(type(phone))

#Checking the length of Dictionary
print(len(phone))

#Access item of dict
print(phone["gagan"])
print(phone["ritika"])
print(phone["dhruv"])

#or get()
print(phone.get("gagan"))

#print keys
print(phone.keys())

#updation value in dict
phone["gagan"]=12334
print(phone)

#add elements in the dict
phone["kia"]=77777
print(phone)

#add new dict to a dict
new_phone={
  "ram":5432
}

phone.update(new_phone)
print(phone)

#remove element from the dict
phone.pop("gagan")
print(phone)

# wants to delete the last time (poptime())
phone.popitem()
print(phone)
```

```
#empty the dict
phone.clear()
print(phone)

#print all of the value of the dict using the loop
phone={
    "gagan":345678,
    "ritika":87654,
    "dhruv":76543,
}

for x in phone:
    print((x))

#printing the keys as well as values
for x,y in phone.items():
    print(x,y)

#nester dict
phone={
    "area1":{
        "x":2,
        "y":4,
        "z":0
    },
    "area2":{
        "a":9,
        "b":3,
        "c":6
    }
}
print(phone["area1"]["y"])
print(phone)

#Question
"""Given a dict in python , write a python program to find the sum of all items in the dict"""
"""
ip=a={
'a':100,
'b':200,
'c':300
}

op=600
"""
dict=a={
'a':100,
'b':200,
'c':300
}
print(sum(dict.values()))

#question 2
dd={
  'x':25,
  'y':18,
  'z':45
}
print(sum(dd.values()))

#zip()
"""
l1=[1,2,3,4,5]
l2=["a","b","c","d","e"]
dict1=dict(zip(li,li))
"""
# l1=[1,2,3,4,5]
# l2=["a","b","c","d","e"]
# dict1=dict(zip(l1, l2))
# print(dict1)

#{1: 'a', 2: 'b', 3: 'c', 4:'d', 5: 'e'}

#question 3
"""Given a string and a number N, we need to mirror the characters from the N-th position up
to the length of the string in alphabetical order.in mirror operation , we change 'a to z' ,
 'boy' and so on"""

input_string=input("enter the string:")
n = int(input("ente the n:"))
#creating dict foe mirror opr.
alphabets="abcdefghijklmnopqrstuvwxyz"
reverse=alphabets[::-1]
dict1=dict(zip(alphabets,reverse))

#finding the part of which do mirror opr
prefix=input_string[0:n-1]
suffix=input_string[n-1:]

#finding the mirror string
mirror=""
for i in range(0,len(suffix)):
    mirror=mirror+dict1[suffix[i]]

#creating final result
res=prefix+mirror
```

```
print(res)
```

string

```
"""
basics
slicing
modifying
concatenation
format
Escape characters
"""
#syntax
"""
using single quotes 'hello world'
using double quotes " I am learning the python"
using the triple quotes ''' my name is gagan '''
"""
#string
"""
It is a sequence of the characters . written in single , double , triple quotes .
It is immutable in nature , but create new string a manipulating the original string
"""
name1='gagandeep singh'
name2='ritika singh'
name3='aryan admi'

print(name1,name2,name3)
print(type(name1))
print(type(name2))
print(type(name3))

#Assigning multi line string to variable
para='''once upon a time there are three cow are line in the
forest one of then are  very clever.'''
  # we use the triple quotes for the multiline
print(para)

#Array-like indexing in string
text="hello, world!"

print(text[0])
print(text[4])
print(text[6])
print(text[-1])

#Traversing a string
```

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

#using list comprehension
lst=[char for char in text]
for i in list:
    print(i)

#Also can find the length of the string
print(len(text))

#Find a char or sub string in the string
print(name1.find('g'))
print(name1.find('a')) # it will give the index of the first occurrence only
print(name1.find('9')) # it will  give the -1 if the letter is not there

#we can also find the substring !
print(text.find('hel')) # it will tell where the starting index of the substring

#find()
"""
return the index of first occurence of the character/substring.
return -1 if not found in original string

"""


#Slicing a string
"""
used to get a part f the string
syntax:
[start:end]
"""
a='gagan'
print(a[2:4])

#Slicing from the start
str='abcdef'
print(str[:3])

#Slicing from the end
print(str[3:])

#Negative indexing
print(str[-3:])
print(str[-3:-1])

#Modifying String
#upper()
```

```
"""
it will convert the string to the upper case
"""
na="gagan"
ma=na.upper()
print(ma)

#lower()
"""
it will convert the string to the lower case
"""
s1=ma.lower()
print(s1)

#capitalize()
"""
It will make the first letter of the string is capital

"""
s2=s1.capitalize()
print(s2)

#strip()
"""
for striping/removeig any trailing whitespaces
"""
str="    hello world!"
print(str.strip())

#replace()
"""
syntax:
str.replace(old_substring,new_string,count)
count ids optional if we do not mention then all occurrence of the substring will be replace
"""
str="kanpur belongs to Delhi to Delhi"
print(str.replace("Delhi","uttar pradesh")) # here we not give the count so it will replace all the
delhi

# but if we given the count then it will replace as per the count
tr="kanpur belongs to Delhi to Delhi"
print(str.replace("Delhi","uttar pradesh",1))

#split()
"""
syntax:
str.split(sep,maxsplit)
used to split the string into the list of the sub string.
```

```
sep,maxsplit--> are the optional parameters
sep(" ")
maxsplit- how many times we want to split at the separator
eg-
"apple are red in colour"
by default it will separate from the space
["apple","are","red","in","colour"]
"""
str="apple are red in colour"
print(str.split())

#giving the separator and maxsplit
str="apple,are,red,in,colour"
print(str.split(",",2))

#concatenation in the string
"""
when we are going to add the two string
"""
str1="hello world!"
str2="how are you"
print(str1+str2)

#format()
"""used to insert variable value in a string

fruit="mango"
fruit="apple"
str="iI have fruits{f1} and {f2}.format(f1=fruit1,f2=fruit2)
"""
fruit="mango"
fruit="apple"
str="I have fruits{f1} and {f2}".format(f1=fruit1,f2=fruit2)
print(str)

#Escape characters
"""
these are some special char some nom printable / reserved character in string"""
#\' single quote  - tom's
#\\ backlash
#/n new line
#\r carriage Return
#\t tap
#\b backspace
#\f form feed
#\ooo octal value
#\xhh hex value
```

```
#Oustion
str="The unexpected always happens"
print(str)
print(len(str))
print(str.find("pp"))
print(str[0:11])
print(str.replace("always","never"))
str1="no matter what" to the string"
print(str+str1)

#WAP to check the given string is palindrome or not
text="mama"
a=text[0:]
if text==a:
    print("The given string is palindrome,")
else:
    print("The given string is not palindrome")

def palindrome(text,a):
   if text==a:
       p="palindrome"
       return p
   else:
       g="string is not palindrome"
       return g
text="mama"
a=text[0:]
print(palindrome(text,a))
```