**PYTHON TRAINING:**

**1)To Print a integer**.

a=10

print(a)

print(id(a))

print(type(a))

**output:**

10

133308638772424

<class 'int'>

**2)Indexing:**

s="Aksh"

print(s[-3])

**output:**

k

**3)Swapping of numbers using (xor) bitwise operations.**

a=10

b=20

a=a^b

b=a^b

a=a^b

print(a)

print(b)

**output:**

20

10

**4) To check a Leap year or not:**

year=int(input("enter the number"))

if(year % 4==0):

print("leap year")

elif(year % 100==0):

print("leap year")

else:

print("is not a leap year")

**Output:**

enter the number2022

is not a leap year.

**Roundoff values:**

**To get the nearest values of the decimal numbers:**

n=float(input("enter the number:"))

print(round(n,-2))

**5) Sum of first n natural numbers:**

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

sum1 = 0

while(n > 0):

sum1=sum1+n

n=n-1

print("The sum of first n natural numbers is",sum1)

**output:**

Enter a number: 10

The sum of first n natural numbers is 55.

**ii)** n=int(input("enter the numbers:"))

sum=0

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

sum +=i

print("sum of first",n,sum)

**output:**

enter the numbers:5

sum of first 5 1

sum of first 5 3

sum of first 5 6

sum of first 5 10

sum of first 5 15

**6) To find a factorial of a given number.**

**i)**n = int(input("Enter any number: "))

f = 1

while n >= 1:

f \*= n

n -= 1

print("Factorial is",f)

**output:**

Enter any number: 6

Factorial is 720

**ii)** n= int(input("enter a number:"))

f=1

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

f\*= i

print("factorial of given number",n,f)

**7)To print the Fibonacci series upto n terms.**

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

a,b = 0, 1

for i in range(n):

print(a, end='')

a ,b = b, a + b

**output:**

enter the number of terms:6

011235

**8) To count digits in a number:**

num=int(input("enter the number:"))

count=0

while num != 0:

num //= 10

count += 1

print("number of digits:", count)

**output:**

enter the number:30

number of digits: 2

**9) to print a table of number:**

num=int(input(“enter a number:”))

print("multiplication table of", num)

for i in range(1,11):

print(num,"x",i,"=",num\*i)

**output:**

enter a number:6

multiplication table of 6

6 x 1 = 6

6 x 2 = 12

6 x 3 = 18

6 x 4 = 24

6 x 5 = 30

6 x 6 = 36

6 x 7 = 42

6 x 8 = 48

6 x 9 = 54

6 x 10 = 60

**10)To print a reverse of a number:**

num=int(input(“enter a number:”))

rev=0

while num > 0:

digit = num % 10

rev = rev \* 10 + digit

num = num // 10

print("reversed number:",rev)

**output:**

enter a number:567

reversed number: 765

**11)To print the prime numbers:**

**BRUTE FORCE METHOD:**

n=int(input("enter the value:"))

x=0

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

x=x^i

print(x)

**output:**

enter the value:6

7

**The output will be in the form of**

**1 1 1**

**2 3 N+1**

**3 0 0**

**4 4 N**

**5 1 1**

**6 7 N+1**

**7 0 0**

**8 8 N**

**For Xor:**

n=int(input("enter the value:"))

if(n%4 == 1):

print(1)

if(n%4 == 2):

print(n+1)

if(n%4 == 3):

print(0)

if(n%4 == 4):

print(n)

**output:**

enter the value:6

7

**ii)** R=int(input("enter the value:"))

L=3

for i in range(L,R+1):

L=L^i

print(L)

**output:**

enter the value:6

7

**iii)** R=int(input("enter the value:"))

L=0

for i in range(3,R+1):

L=L^i

print(L)

**output:**

enter the value:6

4

**iv)** \*\*\*xor of(3^4^5^6)

n=int(input("enter the number"))

x=0

for i in range(3,n+1):

x^=i

print(x)

\*both inputs from user\*

L=int(input("enter the L"))

R=int(input("enter the R"))

x=0

for i in range(L,R+1):

x^=i

print(x)

**o/p:**

enter the L3

enter the R6

3

7

2

4.

**v)** L=int(input("enter the L"))

R=int(input("enter the R"))

x=0

for i in range(L,R+1):

x^=i

print(x)

**output:**

enter the L3

enter the R6

3

7

2

4

**LISTS IN PYTHON:**

Lists is an ordered collection of items/data.

Item could be any type of the data.

EX: L=input().split()

Print(L)

L=[1,2,”AKS”]

Print(L)

Items in list can be accessable the indexing.

List can be deleted by using delete.

**Code:**

L=[1,4,6,30.9,"Akshu"]

print(L)

L.append("dog")

print(L)

L[1]=5

print(L)

del L[2]

print(L)

**output:**

[1, 4, 6, 30.9, 'Akshu']

[1, 4, 6, 30.9, 'Akshu', 'dog']

[1, 5, 6, 30.9, 'Akshu', 'dog']

[1, 5, 30.9, 'Akshu', 'dog']

**ii)** l=[1,2,3,4,5]

for i in range(1,4):

l[i-2]=l[i]

for i in range(0,4):

print(l[i], end = " ")

**output:**

3 4 3 4

**iii)** L=[1,2,3]

L1=L

L[0]=4

print(L1)

**output:**

[4, 2, 3]

**Append:** to add the element in the end of the list.

**Extend:** add all elements of a list to another list.

**Insert:** insert an item at the defined index.

**Example:**

L=[1,2,3,"dog"]

L.append([5,3])

print(L)

L.extend([6,8])

print(L)

L.insert(2,'Cat')

print(L)

**output:**

[1, 2, 3, 'dog', [5, 3]]

[1, 2, 3, 'dog', [5, 3], 6, 8]

[1, 2, 'Cat', 3, 'dog', [5, 3], 6, 8]

**Remove:** removes an item by its value from the list.

**Pop:** removes the last item from the list or removes an item by its index from the list.

**Clear:** removes all items from the list.

**Delete:** it will delete the total list.

**Index:** returns the index of the first matched item.

**Count:** Returns the count of number of items passed as an argument.

**Copy:** returns a copy of the list.

L=[1,2,3,"dog"]

L.remove(2)

print(L)

L.pop()

print(L)

**output:**

[1, 3, 'dog']

[1, 3]

**ii)** L=[15,2,10,5,2,3]

print(L.index(2))

L=[15,2,10,5,2,3]

print(L.count(2))

L=[15,2,10,5,2,3]

print(L.copy())

**output:**

1

2

[15, 2, 10, 5, 2, 3]

**Sort:** sort items in a list in ascending order.

**Reverse:** reverse the order of the items in the list.

L=[15,2,10,5,2,3]

L.sort()

print(L)

L=[15,2,10,5,2,3]

L.reverse()

print(L)

**output:**

[2, 2, 3, 5, 10, 15]

[3, 2, 5, 10, 2, 15]

**Sum:** sum up the numbers in the list.

**Max:** return maximum element of the given list.

**Min:** return minimum element of given list.

L=[15,2,10,5,2,3]

print(sum(L))

print(min(L))

print(max(L))

**output:**

37

2

15

**Length:** returns the length of the list or size of the list.

**Any:** return true if any element of the list is true. If list is empty, return false.

**All:** returns true if all the elements are true or if list is empty.

L=[0,2,0,5,2,3]

print(all(L))

L=[0,2,0,5,2,3]

print(any(L))

L=[15,2,10,5,2,3]

print(len(L))

**output:**

False

True

6

**Example:**

L=[1,2,3,4,5]

L1=['a','hi',4512]

L.append(L1)

print(len(L))

L.extend(L1)

print(len(L))

**output:**

6

9

[1,2,3,4,5,[‘a’,’hi’,4512],’a’,’hi’,4512]

**Membership on list(in, not in):**

**Repetition of list(\*):**

**Concatenation of list(+)**

L=[1,2,3]

L1=[4,5,6]

Print(L+L1)

**output:**

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

**SLICING OF A LIST:**

L[:]-prints all the elements from the list.

L[2:]- prints all the elements from the list starting from index 2.

L[:5]-prints all the elements from the list till the ending index 4.

L[2:6]-prints all the elements from the list from index2 till index 5.

L[-4:-2]-prints all the elements from the list from index -4 till index -2.

L[::2]-prints all the elements from the list with the step 2 from the beginning.

L[::-1]-prints all the elements from the list with step 1 from the last.

L[1:6:3]- prints all the elements from the list from index 1 till index 5 with step 3.

L=[10,11,12,13,14,15,16]

print(L[:])

print(L[2:])

print(L[:5])

print(L[2:6])

print(L[-4:-2])

**output:**

[10, 11, 12, 13, 14, 15, 16]

[12, 13, 14, 15, 16]

[10, 11, 12, 13, 14]

[12, 13, 14, 15]

[13, 14]

**ii)** L=[10,11,12,13,14,15,16]

print(L[::2])

print(L[::-1])

print(L[1:6:3])

**output:**

[10, 12, 14, 16]

[16, 15, 14, 13, 12, 11, 10]

[11, 14]

n1=['python','flask','django','tkinter']

n2=n1

n3=n1[:2]

print(n3)

**output:**

['python', 'flask']

**ii)** n1=['python','flask','django','tkinter']

n2=n1

n3=n1[:2]

n2[0]='scipy'

n3[1]='numpy'

s=10

for i in (n1,n2,n3):

if i[0] == 'python':

s += 1

if i[1] == 'python':

s +=2

print(s)

**output:**

11

**NESTED :**

l=[12,33,'abc',[87,'aks'],[1,2]]

print(l)

print(l[3])

print(l[3][1])

**output:**

[12, 33, 'abc', [87, 'aks'], [1, 2]]

[87, 'aks']

Aks

**LIST COMPREHENSION:**

It is an elegant way to define the and create list in python.

lst=[x \*\* 2 for x in range(1,11) if x%2 == 0]

print(lst)

**output:**

[4, 16, 36, 64, 100]

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

L1=[i for i in L if i%2==0]

print(L1)

**output:**

[2, 4, 6, 8]

Take 2 list and sort the 2 list

L=[1,2,3,6]

L1=[0,5,7]

a=L+L1

print(a)

a.sort()

print(a)

**ii)** a=input().split()

b=inpuy().split()

c=a+b

c=[int(i) for i in c]

print(c)

**input a list and sort the elements in the list based on its length.**

L = input("enter the elements:").split()

L.sort(key=len)

print(L)

**output:**

enter the elements:'hi',1,2,8

["'hi',1,2,8"]

**Read a list and swap first element and the last element in the list.**

L=['hello',12,423,53,0]

L[0],L[-1] = L[-1],L[0]

print(L)

**output:**

[0, 12, 423, 53, 'hello']

L = input("enter the elements:").split()

L[0],L[-1] = L[-1],L[0]

print(L)

**output:**

enter the elements:1 2 3 4 5 6

['6', '2', '3', '4', '5', '1']

**Print the list after deleting the duplicate elements in it.**

a= input("enter the elements:").split()

b=[]

for i in a:

if i not in b:

b.append(i)

print(b)

**output:**

enter the elements:4 a 2 3 6 x

['4', 'a', '2', '3', '6', 'x']

**Print the elements in the list which has occurred odd number of items.**

a= input("enter the elements:").split()

b=[]

for i in a:

if a.count(i)%2 != 0 and i not in b:

b.append(i)

print(b)

**output:**

enter the elements:1 2 3 2 2 5 9 7 7 3

['1', '2', '5', '9']

**Take a nested list and sort the list in ascending to the element in the list.**

**Read a list and print sum of three minimum elements in the list.**

a=list(map(int,input("enter the value:").split()))

a.sort()

print(sum(a[:3]))

**output:**

enter the value:5 6 1 3 8 9 2

6

**Segregate the given list as even elements fist in descending order and then odd elements next in ascending order.**

L=list(map(int,input("enter the value:").split()))

b=[]

L.sort()

print(L)

for i in L:

if i%2 == 0:

b.insert(0,i)

else:

b.append(i)

print(b)

**output:**

enter the value:5 3 2 4 6 7

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

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

**Print product of the elements in the list which are within the given range.**

a=list(map(int,input("enter the value:").split()))

b=[]

m,n=map(int,input("enter the value:").split())

for i in a:

if i in range(m,n+1):

b.append(i)

p=1

for i in b:

for i in b:

p \*= i

print(p)

**Given lists in a list, find the maximum sum of elements of list in a list of lists and prints its index.**

n=int(input("enter the value:"))

a=[map(int,input("enter the value:").split()) for i in range(n)]

m=0

for i in a:

s=sum(i)

if s>m:

m=s

ind=a.index(i)

print(ind)

**TUPLE**: tuple is mutable

t=(1,23,65,12)

print(sum(t))

print(all(t))

print(len(t))

**output:**

101

True

4

**Nested tuple:**

T=(1,2,3.11,(4,5),6,('hi','hello'),7)

print(T)

print(T[3])

print(T[3][0])

**output:**

(1, 2, 3.11, (4, 5), 6, ('hi', 'hello'), 7)

(4, 5)

4

**i) Write a python program to remove an element from a specified index:**

t=tuple(input("enter the value:").split(','))

l=len(t)

print(l)

i=int(input())

t=t[:i] + t[i+1:]

print(t)

**output:**

**To print tuple in a list:**

GFG\_tuple = (1,2,3)

GFG\_list = list(GFG\_tuple)

print(GFG\_tuple)

**output:**

(1, 2, 3)

**Repeated numbers:**

x=tuple(input("enter the value:").split(','))

y=reversed(x)

print(tuple(y))

**output:**

**write a python program to find the repeated items of a tuple:**

t=1,4,5,6,2,3,4,4,7

print(t)

count=t.count(4)

print(count)

**output:**

(1, 4, 5, 6, 2, 3, 4, 4, 7)

3

**Calculate the maximum count:**

my\_tuple=(1,2,3,4,5,4,6,2)

max\_count=max(my\_tuple.count(item) for item in my\_tuple)

print(f"maximum count: {max\_count}")

**output:**

maximum count: 2

**SETS:**

A set is an unordered collection data type that is iterable, mutable and has no duplicate elements.

Sets can be used to perform mathematical set operations like union, intersection, symmetric difference etc.

**Set methods:**

**Add():**adds an element to the set

**Update():**update the set with the union of itself and others.

**Copy():**returns a copy of the set.

**Clear():**removes all elements from the set.

**Remove():**removes an element from the set if the element is not a member raise key error

**Pop():** removes and returns an arbitay set element raise key error if the set is empty

**Discard():**removes an element from the set if it is a member [do nothing if the element is not in set]

**Union:** returns the union of sets in a new set

**Intersection:** Returns the intersection of two sets as a new set.

**Intersection\_update:** updates the set with the intersection of itself and another

s={1,2,3,4,5}

s.add(9)

print(s)

h={88,6,5}

s.update(h)

print(s)

s1=s.copy()

print(s1)

**output:**

{1, 2, 3, 4, 5, 9}

{1, 2, 3, 4, 5, 6, 9, 88}

{1, 2, 3, 4, 5, 6, 9, 88}

**Convert set into the list:**

t={1, 2, 3, 4,4,5,4}

s=t.copy()

print(s)

a\_list=list(s)

print(a\_list)

**output:**

{1, 2, 3, 4, 5}

[1, 2, 3, 4, 5]

s={10,'hi',12,13,3,4}

s.remove(12)

print(s)

s.remove(11)

print(s)

**output:**

{3, 4, 'hi', 10, 13}

a={1, 2, 3, 4,4,5,4}

b={2,3,6,7}

print(a.union(b))

print(a|b)

print(a.intersection(b))

print(a&b)

a.intersection\_update(b)

print(a)

**output:**

{1, 2, 3, 4, 5, 6, 7}

{1, 2, 3, 4, 5, 6, 7}

{2, 3}

{2, 3}

{2, 3}

a={1, 2, 3, 4,5}

b={2,3,6,7}

c={1,2,3,8}

print(a.union(b))

print(a|b)

print(a.intersection(b))

print(a&b)

a.intersection\_update(b)

b.intersection\_update(c)

print(c)

print(a)

**output:**

{1, 2, 3, 4, 5, 6, 7}

{1, 2, 3, 4, 5, 6, 7}

{2, 3}

{2, 3}

{2, 3}

{2, 3}

**Difference:** returns the difference of 2 or more sets as a new set.

**Difference\_update:** removes all elements of another set from this set.

**Symmetric\_difference:** returns the symmetric difference of two sets as a new set.

**Isdisjoint:**returns true if two sets have a null intersection.

**Issubset:** returns true if another set contains this set.

**Issuperset:** returns true if this set contains another set.

a={1,2,3,4,5}

b={3,9,44,6,5}

print(a.isdisjoint(b))

c={1,2,3,4}

d={2,3}

print(c.issuperset(d))

print(d.issubset(c))

**output:**

False

True

True

**Set built in functions:**

**Sum:** returns the sum of all the elements in the set.

**Max:** return the largest item in the set:

**Min:** returns the smallest item in the set.

**Len:** return the length the number of items in the set.

a={1,2,3,4,5}

b={3,9,44,6,5}

print(sum(a))

print(min(b))

print(max(a))

print(len(a))

print(len(b))

output:

15

3

5

5

5

**All:** return true if all the elements of the set are true(or if the set is empty).

**Any:** return true if any element of the set is true. if the set is empty, return false:

**Sorted:** return a new sorted list from elements in the set (does not sorted the set itself).

a={1,0,3,4,5}

print(all(a))

print(any(a))

print(sorted(a))

**Output:**

False

True

[0, 1, 3, 4, 5]

a=[1,1,2,2,2,5,7,7,5,3,0]

print(sorted(a))

**output:**

[0, 1, 1, 2, 2, 2, 3, 5, 5, 7, 7]

**Conversion of set into list and sorting the elements:**

t={1, 2, 3, 4,4,11,5,4,0,7,1,}

s=t.copy()

print(s)

a\_list=list(s)

print(a\_list)

print(sorted(t))

**output:**

{0, 1, 2, 3, 4, 5, 7, 11}

[0, 1, 2, 3, 4, 5, 7, 11]

[0, 1, 2, 3, 4, 5, 7, 11]

**operations on set:**

membership on tuple(in,not in)

equal(==),not equal(!=)

subset(<=),proper subset(<)

superset(>=), proper superset(>)

union(|), intersection(&)

difference(-), symmetric\_difference(^)

a={1,2,3}

print(1 in a)

print(2 not in a)

c={1,2,3}

d={1,2,3}

print(c==d)

print(c!=d)

**output:**

True

False

True

False

**FROZEN SET:**

Frozensets are immutable sets.

Frozensets can be created using the function frozenset().

This datatype supports methods like copy()

set1={10,20,30,40,50}

set2={60,70,10,30,40,80,20,50}

print(set1.issubset(set2))

print(set2.issuperset(set1))

**output:**

True

True

**Set comprehension:**

Returns a set based on existing iterables.

Syntax:{expression(variable) for variable in input\_set[predicate][, …..]}

res={s for s in [1,2,3] if s % 2}

print(res)

**output:**

{1, 3}

**To print even numbers:**

res={s for s in [1,2,3,6,8] if s % 2 == 0}

print(res)

**output:**

{8, 2, 6}

In sorted order:

res={s for s in [1,2,3,6,8] if s % 2 == 0}

print(sorted(res))

**output:**

{2,6,8}

**DICTIONARY:**

A dictionary is a collection which is unsorted, changeable and indexed.

In python dictionaries are written with curly brackets, and they have keys and values.

It’s a key value pair.

**Creating a dictionary:**

Values can be of any datatype, and can be duplicated.

Whereas keys cannot be repeated and must be immutable.

**Ex:**

D={}# **empty dictionary**

Print(D)

Print(type(D))

1. d={}

for i in range(3):

key=input("key:")

value=input("value:")

d[key]=value

print(d)

**Accessing the elements in a dictionary:**

i)d={1:'hi','a':123,100:32.4}

for i in d:

print(i,":",d[i])

**output:**

1 : hi

a : 123

100:32.4

ii)d={1:'hi','a':123,100:32.4}

print(d.get('a'))

**output:**

123

**Modify and delete a dictionary:**

Only values in dictionary can be changed but keys should be immutable.

d={1:'hi','a':123,100:32.4}

d[1]=1000

print(d)

**output:**

{1: 1000, 'a': 123, 100: 32.4}

Adding an element to the dictionary:

d={1:'hi','a':123,100:32.4}

d[2]='python'

print(d)

**output:**

{1: 'hi', 'a': 123, 100: 32.4, 2: 'python'}

**Get:** it is the conventional method to access a value for a key.

**Update:** adds dictionary dict2’s key-values pairs to dict.

**Copy:** they copy method returns

**Pop:** removes and returns an element from a dictionary having the given key.

**Popitem:** removes the arbitrary key-value pair from the dictionary and returns is as tuple.

**Clear:** the clear method removes all items from the dictionary.

d={1:'hi','a':123,100:32.4}

d.pop('a')

print(d)

d.popitem()

print(d)

d.clear()

print(d)

**output:**

{1: 'hi', 100: 32.4}

{1: 'hi'}

{}

**Keys:** returns list of dictionary dict’s keys.

**Values:** returns a list of all the values available in a given dictionary.

**Items:** returns a list of dict’s (key, value) tuple pairs.

keys():

d={1:'hi','a':123,100:32.4}

print(d.keys())

O/P:

dict\_keys([1, 'a', 100])

values():

d={1:'hi','a':123,100:32.4}

print(d.values())

O/P:

dict\_values(['hi', 123, 32.4])

items():

d={1:'hi','a':123,100:32.4}

print(d.items())

O/P:

dict\_items([(1, 'hi'), ('a', 123), (100, 32.4)])

**Fromkeys**: create a new dictionary with the keys from sequence

and values set to value.

**Setdefault():** Set dict[key]=default if key is not already in dict.

c={1:'hi','a':123,100:32.4}

print(c.setdefault('a'))

print(c.setdefault('b'))

print(c)

print(c.setdefault('d',200))

print(c)

**output:**

123

None

{1: 'hi', 'a': 123, 100: 32.4, 'b': None}

200

{1: 'hi', 'a': 123, 100: 32.4, 'b': None, 'd': 200}

**NESTED DICTIONARY:**

Dictionary inside another dictionary is nested dictionary:

d={1:'hi','a':123,100:{2:'abc','x':452,2.3:120},5.4:'python'}

print(d[100]['x'])

**output:**

452

**Ex:** student = {

"name": "emma",

"class": 9,

"marks": 75

}

print(student)

**output:**

{'name': 'emma', 'class': 9, 'marks': 75}

**DICTIONARY COMPREHENSION:**

Dictionary comprehension is a method for transforming a one dictionary into another dictionary.

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

d1={k:v\*2 for (k,v) in d.items()}

print(d1)

**output:**

{'a': 2, 'b': 4, 'c': 6, 'd': 8, 'e': 10}

**PROBLEMS:**

**1)Write a python program to add a key to a dictionary.**

import operator

d={}

n=int(input("no.of elements:"))

for i in range(n):

k=input("enter the key for first dict:")

v=input("enter the value:")

d[k]=v

key=input()

value=input()

d.update({key:value})

print(d)

**output:**

no.of elements:4

enter the key for first dict:1

enter the value:3

enter the key for first dict:6

enter the value:9

enter the key for first dict:3

enter the value:1

enter the key for first dict:3

enter the value:0

{'1': '3', '6': '9', '3': '0', '': ''}

**2) write a python program to check whether a given key already exists in a dictionary.**

import operator

d={}

n=int(input("no.of elements:"))

for i in range(n):

k=input("enter the key for dict:")

v=input("enter the value:")

d[k]=v

key=input()

if key in d:

print('key is present in the dictionary')

else:

print('key is not present in the dictionary')

**output:**

no.of elements:2

enter the key for dict:3

enter the value:5

enter the key for dict:1

enter the value:5

key is not present in the dictionary

**3)write a python program to generate and print a dictionary that contains a number (between 1 and n) in the form (x and x \* x).**

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

d=dict()

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

d[x]=x\*x

print(d)

**output:**

input a number:4

{1: 1, 2: 4, 3: 9, 4: 16}

**4)Write a python program to merge two python dictionaries.**

d={}

d1={}

n=int(input("no of elements:"))

for i in range(n):

k=input("enter the key for dict:")

v=input("enter its value:")

d[k]=v

for i in range(n):

k=input("enter the key for dict:")

v=input("enter its value:")

d1[k]=v

d2=d.copy()

d2.update(d1)

print(d2)

**output:**

no of elements:3

enter the key for dict:12

enter its value:13

enter the key for dict:16

enter its value:18

enter the key for dict:12

enter its value:90

enter the key for dict:87

enter its value:65

enter the key for dict:43

enter its value:42

enter the key for dict:77

enter its value:32

{'12': '90', '16': '18', '87': '65', '43': '42', '77': '32'}

**5)write a python program to remove a key from a dictionary.**

d={}

n = int(input("no of elements:"))

for i in range(n):

k = int(input("enter key for dict:"))

v = int(input("enter its value:"))

d[k]=v

key=int(input())

if key in d:

del d[key]

print(d)

**output:**

no of elements:2

enter key for dict:5

enter its value:1

enter key for dict:6

enter its value:2

5

{6: 2}

**6)write a python program to map two lists into a dictionary.**

keys=input("enter the values:").split()

values=input("enter the values:").split()

d=dict(zip(keys, values))

print(d)

**output:**

enter the values:1 2 3 4

enter the values:7 8 3 9

{'1': '7', '2': '8', '3': '3', '4': '9'}

**ii)** l1=[1,2,3]

l2=[5,6,7]

d={}

for i in range(0,3):

key=l1[i]

value=l2[i]

d[key]=value

print(d)

**output:**

{1: 5, 2: 6, 3: 7}

**7)write a python program to combine two dictionary adding values for the common keys.**

n1=int(input())

d1={}

for i in range(n1):

k\_v\_pair=tuple(map(int,input().split(":")))

d1[k\_v\_pair[0]]=k\_v\_pair[1]

print("d1 : ",d1)

n2=int(input())

d2={}

for i in range(n2):

k\_v\_pair=tuple(map(int,input().split(":")))

d2[k\_v\_pair[0]]=k\_v\_pair[1]

d3=d1.copy()

for k,v in d2.items():

if k in d3:

d3[k]+=v

else:

d3[k]=v

print(d3)

**output:**

**ii)** from collections import Counter

d={}

d1={}

n=int(input("no.of elements:"))

for i in range(n):

k=input("enter the key for first dict:")

v=input("enter its value:")

d[k]=v

for i in range(n):

k=input("enter the key for second dict:")

v=input("enter its value:")

d1[k]=v

d2 = Counter(d) + Counter(d1)

print(d2)

**iii)** dict1={'a':10,'b':20,'c':30}

dict2={'b':5,'c':15,'d':25}

combined\_dict={}

for key in dict1:

if key in dict2:

combined\_dict[key]=dict1[key]+dict2[key]

else:

combined\_dict[key]=dict1[key]

for key in dict2:

if key not in combined\_dict:

combined\_dict[key]=dict2[key]

print("Combined Dictionary:",combined\_dict)

**output:**

Combined Dictionary: {'a': 10, 'b': 25, 'c': 45, 'd': 25}

**STRING:**

**Strings:** strings are immutable, hence elements of a string cannot be changed once it has been assigned.

Only new strings can be reassigned to the

We can directly delete a string using del keyword.

**String methods and functions:**

**String.ascii\_letters:** Concatation of the ascii\_lowercase and ascii\_uppercase constants.

**String.ascii\_lowercase:** Concatenation of lowercase letters.

**String.ascii\_uppercase:** concatenation of uppercase letters.

**String.punctuations:** ASCII characters having punctuation characters.

import string

print(string.ascii\_letters)

print(string.ascii\_lowercase)

print(string.ascii\_uppercase)

print(string.punctuation)

**output:**

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

ABCDEFGHIJKLMNOPQRSTUVWXYZ

!"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~

**String.digits:** digit in strings.

**String.hexdigits:** hexadigit in strings.

**String.octdigits**: octadigit in a sting

import string

print(string.digits)

print(string.hexdigits)

print(string.octdigits)

**output:**

0123456789

0123456789abcdefABCDEF

01234567

**String.endswitch**: returns true if a string ends with the given suffix otherwise returns false

**String.startswith:** returns true if a string with the given prefix otherwise returns false.

**Replace:** returns a copy of he sting where all occurrences of a substring is replaced with another substring.

print('hello'.endswith('o'))

print('hello'.startswith('h'))

print('hello'.replace('l','t'))

**output:**

True

True

Hetto

**String.isdigit:** returns “true” if all characters in the string are digits, otherwise, it returns “False”.

**String.isalpha:** returns “true” if all characters in the string are alphabets, otherwise it returns “false”.

**String.isdecimal:** returns true if all characters in a string are decimal.

print('100'.isdigit())

print('abc'.isalpha())

print('72'.isdecimal())

**output:**

True

True

True

**String.isalnum:** returns true if all the characters in a given string are alphanumeric

**String.istitle:** returns true if the string is a titlecased string.

print('ab12'.isalnum())

print('Hello World'.istitle())

**output:**

True

True

**String,partition:** splits the string at the first occurrence of the separator and returns tuple.

**String.index:** returns the position of the first occurrence of substring in a string.

**String.rindex:** returns the highest index of the substring inside the string if the substring is found.

**String.splitlines:** returns a lsit of lines in the string.

print('hello'.upper())

print('HELLo'.lower())

print('hElLo'.swapcase())

**output:**

HELLO

hello

HeLlO

**1)**str2="""hi

this

is sash"""

print(str2.splitlines())

**output:**

['hi', 'this', 'is sash']

**2)** str="potti"

print(str.index('t'))

print(str.rindex('i'))

**output:**

2

4

**String.capitalize:** return the word with its first character capitalized.

**String.find:** return the lowest index in a sub string.

**String.rfinf:** find the highest index.

**String.count:** returns the number of (non- overlapping) occurrences of substring sub in string.

print('python programming'.capitalize())

print('python cython'.find('th'))

print('python cython'.rfind('th'))

print('python cython'.count('th'))

**output:**

Python programming

2

9

2

**Len:** returns the length of the string.

**Max:** returns the highest alphabetical character in a string

**Min:** returns the minimum alphabetical character in a string.

print('python programming'.capitalize())

print('python cython'.find('th'))

print('python cython'.rfind('th'))

print('python cython'.count('th'))

**output:**

Python programming

2

9

2

**Slicing of a string:**

**S[:]:** prints all the elements from the string.

**S[2:]:** prints all the elements from the string starting from index 2.

**S[:5]:** prints all the elements from the string till the ending index 4.

**S[2:6]:** prints the elements from the string from index 2 till index 5.

**S[-4:2]:** prints the elements from the string from index -4 till index -3.

**S[::2]:** prints all the elements from the string with the step 2 from the beginning.

**S[::-1]:** prints all the elements from the string with step 1 from the last.

s='pranathi'

print(s[:])

print(s[2:])

print(s[:6])

print(s[2:4])

print(s[-3:-5])

**output:**

pranathi

anathi

pranat

an

s='python'

print(s[::])

print(s[1::4])

**output:**

python

yn

**3)**str1="This is 2020"

str2="2020"

print(str1.isdigit())

print(str2.isdigit())

**output:**

False

True

**Problem statement:**

**1)write a python program to calculate the length of a string.**

def len1(str1):

count=0

for char in str1:

count += 1

return count

str1=input("enter the value:")

print(len1(str1))

**output:**

enter the value:coder123

8

**2)WRITE A PYTHON SCRIPT THAT TAKES INPUT FROM THR USER AND DISPLAYS THE INPUT BACK IN UPPER CASE AND LOWER CASE.**

s=input("enter the word:")

print(s.upper())

print(s.lower())

**output:**

enter the word: python

PYTHON

Python

**ii)** str=input("enter the value:")

print("your input in upper case",str.upper())

print("your input in upper case",str.lower())

**output:**

enter the value:akshaya

your input in upper case AKSHAYA

your input in upper case Akshaya

**iii**) s1="my car is BMW"

s2="BMW"

s3=s1.replace(s2,s2.upper())

print(s3)

**output:**

my car is BMW.

**iv)** def swap(str1):

res=""

for i in str1:

if i.isupper():

res +=i.lower()

else:

res +=i.upper()

return res

str1=input("enter the value:")

print(swap(str1))

**output:**

enter the value: akshaya

AKSHAYA

**4) write a python program to remove all consecutive duplicates from a given string.**

str=input("Enter the number:")

result=str[0]

for i in range (1,len(str)):

if str[i] !=result[-1]:

result += str[i]

print(result)

**Output:**

Enter the number:58555876

585876

**5)write a python program to move all spaces to the front of a given string in single traversal.**

s=input("enter the value:")

spaces = ""

result = ""

for char in s:

if char == " ":

spaces += char

else:

result += char

final\_string = spaces + result

print(final\_string)

**output:**

enter the value:akshaya pallamaina

akshayapallamaina

**6) write a python program to create a string from two strings concatenating uncommon characters of the said strings.**

str1="hello"

str2="world"

set1=set(str1)

set2=set(str2)

uncommon\_chars=set2.symmetric\_difference(set1)

print(uncommon\_chars)

uncommon\_string=''.join(uncommon\_chars)

print(uncommon\_string)

**output:**

{'w', 'd', 'e', 'h', 'r'}

wdehr

**ii)** s1="hello"

s2="world"

uncommon\_chars = "".join(set(s1) ^ set(s2))

print(uncommon\_chars)

**output:**

rwhde

**7) write a python program to find the maximum occurring character in a given string.**

s=input("enter the value:")

max\_char = max(s, key=s.count)

print(f"Maximum occurring character: {max\_char}")

**output:**

enter the value:Treat

Maximum occurring character: T

**PYTHON FUNCTION DECLEARTION:**

Types of functions in python:

**Built-in library function:** These are standard function in Python that are available to use.

**User-defined function:** We can create a own functions based on our requirements.

def fun():

print("welcome to GFG")

fun()

**output:**

welcome to GFG

**ii)** def add(num1:int , num2: int) -> int:

num3 = num1 + num2

return num3

num1, num2 = 5,13

ans = add(num1, num2)

print(f"the addition of {num1} and {num2} results {ans}.")

**output:**

the addition of 5 and 13 results 18.

**To check the number is a prime or not:**

def check\_prime():

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

if num > 1:

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

if num % i == 0:

print(num, "is not a prime number")

break

else:

print(num,"is a prime number")

else:

print(num, "is not a prime number")

check\_prime()

**output:**

enter a number:30

30 is not a prime number

**Arguments:**

def myfun(x, y=50):

print("x:",x)

print("y:",y)

myfun(10)

**output:**

x: 10

y: 50

**to demonstrate keyword arguments:**

def student(firstname, lastname):

print(firstname, lastname)

student(firstname='akshaya', lastname='pallamaina')

student(lastname='sravanthi',firstname='baikani')

**output:**

akshaya pallamaina

baikani sravanthi

**positional arguments:**

def nameAge(name, age):

print("Hi, I am ", name)

print("My age is",age)

print("case-1:")

print("Akshaya",20)

print("/ncase-2:")

nameAge(20, "Akshaya")

**output:**

case-1:

Akshaya 20

/ncase-2:

Hi, I am 20

My age is Akshaya

**ii)** def my\_sum(\*args):

result = 0

for x in args:

result += x

return result

print(my\_sum(1,2,3))

**output:**

6

**Unpacking operator(\*\*):**

def concatenate(\*\*words):

result = ""

for arg in words.values():

result += arg

return result

print(concatenate(a="real", b="python", c="is", d="great", e="!"))

**output:**

realpythonisgreat!

**To print a factorial of a number recursively:**

def recursive\_factorial(n):

if n == 1:

return n

else:

return n \* recursive\_factorial(n-1)

num=6

if num < 0:

print("Invalid input ! please enter a positive number.")

elif num == 0:

print("Fcatorial of number is 0 is 1")

else:

print("Factorial of number", num, "=", recursive\_factorial(num))

**output:**

Factorial of number 6 = 720

Recursion: Calling a function inside itself:

**To print factorial value in reverse using recursive function (5\*4\*3\*2\*1):**

def factorial\_expression(n):

components = []

factorial = 1

for i in range(n, 0, -1):

components.append(str(i))

factorial \*= i

expression = '\*'.join(components)

print(f"The factorial of {n} is: {expression}")

print(f"Which equals: {factorial}")

factorial\_expression(5)

**output:**

The factorial of 5 is: 5\*4\*3\*2\*1

Which equals: 120

**1)**def fun(n):

if(n>0):

print(n)

fun(n-1)

n=int(input("enter the value:"))

fun(n)

**Output:**

enter the value:5

5

4

3

2

1

**2)** def fun(n):

if(n>0):

fun(n-1)

print(n)

n=int(input("enter the value:"))

fun(n)

**output:**

enter the value:4

1

2

3

4

**Fibonacci series using recursion:**

def fibonacci(n):

if n <= 1:

return n

else:

return fibonacci(n-1) + fibonacci(n-2)

num\_terms = int(input("enter the no of terms:"))

for i in range(num\_terms):

print(f"fibonacci({i}) = {fibonacci(i)}")

**output:**

enter the no of terms:5

fibonacci(0) = 0

fibonacci(1) = 1

fibonacci(2) = 1

fibonacci(3) = 2

fibonacci(4) = 3

**CODEFORCES PROBLEMS:**

**Codeforces:**

**1)**One hot summer day Pete and his friend Billy decided to buy a watermelon. They chose the biggest and the ripest one, in their opinion. After that the watermelon was weighed, and the scales showed *w* kilos. They rushed home, dying of thirst, and decided to divide the berry, however they faced a hard problem.

Pete and Billy are great fans of even numbers, that's why they want to divide the watermelon in such a way that each of the two parts weighs even number of kilos, at the same time it is not obligatory that the parts are equal. The boys are extremely tired and want to start their meal as soon as possible, that's why you should help them and find out, if they can divide the watermelon in the way they want. For sure, each of them should get a part of positive weight.

**Code:**

w=int(input("enter the weight of the watermelon:"))

if (w%2 != 0): (if(w%2 !=0 & w==2): for two) number.

print("no")

else:

x=w/2

if(x%2 == 0):

print("weight of the 2 pieces",x,x)

else:

print("weight of the 2 pieces",x-1,x+1)

**output:**

enter the weight of the watermelon:6

weight of the 2 pieces 2.0 4.0

**2) Codeforces:617**

An elephant decided to visit his friend. It turned out that the elephant's house is located at point 0 and his friend's house is located at point *x*(*x* > 0) of the coordinate line. In one step the elephant can move 1, 2, 3, 4 or 5 positions forward. Determine, what is the minimum number of steps he need to make in order to get to his friend's house.

**Code:**

x=int(input("enter the number:"))

if(x<5):

print(1)

if(x%5==0):

print(x/5)

else:

print(x//5+1)

**output:**

i)enter the number:4

1

1

ii) enter the number:5

1.0