In [1]:

```
print('Hello, world!')
```

```
Hello, world!
```

In [2]:

```
n=5
print('no is =',n)
```

```
no is = 5
```

In [4]:

```
num=8
num_sqrt = num ** 0.5
num_sqrt
```

Out[4]:

```
2.8284271247461903
```

In [5]:

```
x = input('Enter value of x: ')
y = input('Enter value of y: ')
temp = x
x = y
y = temp
print('no is',x,y)
```

```
Enter value of x: 8
Enter value of y: 9
no is 9 8
```

In [6]:

```
num = int(input("Enter a number: "))
if (num % 2) == 0:
   print("{0} is Even".format(num))
else:
   print("{0} is Odd".format(num))
```

```
Enter a number: 4
4 is Even
```

In [7]:

```
year = 2000

# To get year (integer input) from the user
# year = int(input("Enter a year: "))

if (year % 4) == 0:
   if (year % 100) == 0:
       if (year % 400) == 0:
           print("{0} is a leap year".format(year))
       else:
           print("{0} is not a leap year".format(year))
   else:
       print("{0} is a leap year".format(year))
else:
   print("{0} is not a leap year".format(year))
```

```
2000 is a leap year
```

In [8]:

```
num1 = 10
num2 = 14
num3 = 12
if (num1 >= num2) and (num1 >= num3):
   largest = num1
elif (num2 >= num1) and (num2 >= num3):
   largest = num2
else:
   largest = num3

print("The largest number between",num1,",",num2,"and",num3,"is",largest)
```

```
The largest number between 10 , 14 and 12 is 14
```

In [9]:

```
if num > 1:
   # check for factors
   for i in range(2,num):
       if (num % i) == 0:
           print(num,"is not a prime number")
           print(i,"times",num//i,"is",num)
           break
   else:
       print(num,"is a prime number")

# if input number is less than
# or equal to 1, it is not prime
else:
   print(num,"is not a prime number")
```

```
4 is not a prime number
2 times 2 is 4
```

In [10]:

```
lower = 900
upper = 1000

print("Prime numbers between",lower,"and",upper,"are:")

for num in range(lower,upper + 1):
   # prime numbers are greater than 1
   if num > 1:
       for i in range(2,num):
           if (num % i) == 0:
               break
       else:
           print(num)
```

```
Prime numbers between 900 and 1000 are:
907
911
919
929
937
941
947
953
967
971
977
983
991
997
```

In [11]:

```
num = 7
factorial = 1

# check if the number is negative, positive or zero
if num < 0:
   print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
   print("The factorial of 0 is 1")
else:
   for i in range(1,num + 1):
       factorial = factorial*i
   print("The factorial of",num,"is",factorial)
```

```
The factorial of 7 is 5040
```

In [12]:

```
nterms = 10

# uncomment to take input from the user
#nterms = int(input("How many terms? "))

# first two terms
n1 = 0
n2 = 1
count = 0

# check if the number of terms is valid
if nterms <= 0:
   print("Please enter a positive integer")
elif nterms == 1:
   print("Fibonacci sequence upto",nterms,":")
   print(n1)
else:
   print("Fibonacci sequence upto",nterms,":")
   while count < nterms:
       print(n1,end=' , ')
       nth = n1 + n2
       # update values
       n1 = n2
       n2 = nth
       count += 1
```

```
Fibonacci sequence upto 10 :
0 , 1 , 1 , 2 , 3 , 5 , 8 , 13 , 21 , 34 ,
```

In [14]:

```
 #Python program to check if the number provided by the user is an Armstrong number or not
# take input from the user
num = int(input("Enter a number: "))
# initialize sum
sum = 0
# find the sum of the cube of each digit
temp = num
while temp > 0:
   digit = temp % 10
   sum += digit ** 3
   temp //= 10
# display the result
if num == sum:
   print(num,"is an Armstrong number")
else:
   print(num,"is not an Armstrong number")
```

```
Enter a number: 585
585 is not an Armstrong number
```

In [15]:

```
lower = 100
upper = 2000

# To take input from the user
# lower = int(input("Enter lower range: "))
# upper = int(input("Enter upper range: "))

for num in range(lower, upper + 1):

   # order of number
   order = len(str(num))

   # initialize sum
   sum = 0

   # find the sum of the cube of each digit
   temp = num
   while temp > 0:
       digit = temp % 10
       sum += digit ** order
       temp //= 10

   if num == sum:
       print(num)
```

```
153
370
371
407
1634
```

In [16]:

```
# Python program to find the sum of natural numbers up to n where n is provided by user

# change this value for a different result
num = 16

# uncomment to take input from the user
#num = int(input("Enter a number: "))

if num < 0:
   print("Enter a positive number")
else:
   sum = 0
   # use while loop to iterate un till zero
   while(num > 0):
       sum += num
       num -= 1
   print("The sum is",sum)
```

```
The sum is 136
```

In [17]:

```
# Python Program to find numbers divisible by thirteen from a list using anonymous function

# Take a list of numbers
my_list = [12, 65, 54, 39, 102, 339, 221,]

# use anonymous function to filter
result = list(filter(lambda x: (x % 13 == 0), my_list))

# display the result
print("Numbers divisible by 13 are",result)
```

```
Numbers divisible by 13 are [65, 39, 221]
```

In [18]:

```
# Python program to convert decimal number into binary, octal and hexadecimal number system

# Change this line for a different result
dec = 344

print("The decimal value of",dec,"is:")
print(bin(dec),"in binary.")
print(oct(dec),"in octal.")
print(hex(dec),"in hexadecimal.")
```

```
The decimal value of 344 is:
0b101011000 in binary.
0o530 in octal.
0x158 in hexadecimal.
```

In [19]:

```
def computeHCF(x, y):

# choose the smaller number
    if x > y:
        smaller = y
    else:
        smaller = x
    for i in range(1, smaller+1):
        if((x % i == 0) and (y % i == 0)):
            hcf = i

    return hcf

num1 = 54
num2 = 24

# take input from the user
# num1 = int(input("Enter first number: "))
# num2 = int(input("Enter second number: "))

print("The H.C.F. of", num1,"and", num2,"is", computeHCF(num1, num2))
```

```
The H.C.F. of 54 and 24 is 6
```

In [20]:

```
# Python Program to find the L.C.M. of two input number

# define a function
def lcm(x, y):
   """This function takes two
   integers and returns the L.C.M."""

   # choose the greater number
   if x > y:
       greater = x
   else:
       greater = y

   while(True):
       if((greater % x == 0) and (greater % y == 0)):
           lcm = greater
           break
       greater += 1

   return lcm

# change the values of num1 and num2 for a different result
num1 = 54
num2 = 24

# uncomment the following lines to take input from the user
#num1 = int(input("Enter first number: "))
#num2 = int(input("Enter second number: "))

print("The L.C.M. of", num1,"and", num2,"is", lcm(num1, num2))
```

```
The L.C.M. of 54 and 24 is 216
```

In [21]:

```
# Python Program to find the factors of a number

# define a function
def print_factors(x):
   # This function takes a number and prints the factors

   print("The factors of",x,"are:")
   for i in range(1, x + 1):
       if x % i == 0:
           print(i)

# change this value for a different result.
num = 320

# uncomment the following line to take input from the user
#num = int(input("Enter a number: "))

print_factors(num)
```

```
The factors of 320 are:
1
2
4
5
8
10
16
20
32
40
64
80
160
320
```

In [22]:

```
# Python program to display the Fibonacci sequence up to n-th term using recursive function

def recur_fibo(n):
   """Recursive function to
   print Fibonacci sequence"""
   if n <= 1:
       return n
   else:
       return(recur_fibo(n-1) + recur_fibo(n-2))

# Change this value for a different result
nterms = 10

# uncomment to take input from the user
#nterms = int(input("How many terms? "))

# check if the number of terms is valid
if nterms <= 0:
   print("Plese enter a positive integer")
else:
   print("Fibonacci sequence:")
   for i in range(nterms):
       print(recur_fibo(i))
```

```
Fibonacci sequence:
0
1
1
2
3
5
8
13
21
34
```

In [23]:

```
# Python program to find the factorial of a number using recursion

def recur_factorial(n):
   """Function to return the factorial
   of a number using recursion"""
   if n == 1:
       return n
   else:
       return n*recur_factorial(n-1)

# Change this value for a different result
num = 7

# uncomment to take input from the user
#num = int(input("Enter a number: "))

# check is the number is negative
if num < 0:
   print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
   print("The factorial of 0 is 1")
else:
   print("The factorial of",num,"is",recur_factorial(num))
```

```
The factorial of 7 is 5040
```

In [24]:

```
# Program to check if a string
#  is palindrome or not

# change this value for a different output
my_str = 'aIbohPhoBiA'

# make it suitable for caseless comparison
my_str = my_str.casefold()

# reverse the string
rev_str = reversed(my_str)

# check if the string is equal to its reverse
if list(my_str) == list(rev_str):
   print("It is palindrome")
else:
   print("It is not palindrome")
```

```
It is palindrome
```

In [25]:

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

my_str = "Hello!!!, he said ---and went."

# To take input from the user
# my_str = input("Enter a string: ")

# remove punctuation from the string
no_punct = ""
for char in my_str:
   if char not in punctuations:
       no_punct = no_punct + char

# display the unpunctuated string
print(no_punct)
```

```
Hello he said and went
```

In [26]:

```
# Program to sort alphabetically the words form a string provided by the user

# change this value for a different result
my_str = "Hello this Is an Example With cased letters"

# uncomment to take input from the user
#my_str = input("Enter a string: ")

# breakdown the string into a list of words
words = my_str.split()

# sort the list
words.sort()

# display the sorted words

print("The sorted words are:")
for word in words:
   print(word)
```

```
The sorted words are:
Example
Hello
Is
With
an
cased
letters
this
```

In [27]:

```
# Program to count the number of each vowel in a string

# string of vowels
vowels = 'aeiou'

# change this value for a different result
ip_str = 'Hello, have you tried our turorial section yet?'

# uncomment to take input from the user
#ip_str = input("Enter a string: ")

# make it suitable for caseless comparisions
ip_str = ip_str.casefold()

# make a dictionary with each vowel a key and value 0
count = {}.fromkeys(vowels,0)

# count the vowels
for char in ip_str:
   if char in count:
       count[char] += 1

print(count)
```

```
{'a': 2, 'e': 5, 'i': 3, 'o': 5, 'u': 3}
```

In [28]:

```
Example: Calculate Average of Numbers Using Arrays
#include <iostream>
using namespace std;
int main()
{
    int n, i;
    float num[100], sum=0.0, average;
    cout << "Enter the numbers of data: ";
    cin >> n;
    while (n > 100 || n <= 0)
    {
        cout << "Error! number should in range of (1 to 100)." << endl;
        cout << "Enter the number again: ";
        cin >> n;
    }
    for(i = 0; i < n; ++i)
    {
        cout << i + 1 << ". Enter number: ";
        cin >> num[i];
        sum += num[i];
    }
    average = sum / n;
    cout << "Average = " << average;
    return 0;
}
```

```
  File "<ipython-input-28-6ecefa0406d7>", line 1
    Example: Calculate Average of Numbers Using Arrays
                       ^
SyntaxError: invalid syntax
```

In [13]:

```
#Listand tuple ka b same

 #Updating list
list = ['physics', 'chemistry', 1997, 2000];
print (list[2])
list[2] = 2001
print ("New value available at index 2 : ")
print (list[2])

#Delete List Elements
list1 = ['physics', 'chemistry', 1997, 2000];
print (list1)
del list1[2];
print ("After deleting value at index 2 : ")
print (list1)

#basic operation
print(len([1, 2, 3]))
print([1, 2, 3] + [4, 5, 6])
print(['Hi!'] * 4)
print(3 in [1, 2, 3])
for x in [1, 2, 3]: print (x),

    #slicing
L = ['spam', 'Spam', 'SPAM!']
L[1:]
```

```
1997
New value available at index 2 :
2001
['physics', 'chemistry', 1997, 2000]
After deleting value at index 2 :
['physics', 'chemistry', 2000]
3
[1, 2, 3, 4, 5, 6]
['Hi!', 'Hi!', 'Hi!', 'Hi!']
True
1
2
3
```

Out[13]:

```
['Spam', 'SPAM!']
```

In [ ]:

```
cmp(list1, list2)
Compares elements of both lists.

2   len(list)
Gives the total length of the list.

3   max(list)
Returns item from the list with max value.

4   min(list)
Returns item from the list with min value.

5   list(seq)
Converts a tuple into list.
list.append(obj)
Appends object obj to list

2   list.count(obj)
Returns count of how many times obj occurs in list

3   list.extend(seq)
Appends the contents of seq to list

4   list.index(obj)
Returns the lowest index in list that obj appears

5   list.insert(index, obj)
Inserts object obj into list at offset index

6   list.pop(obj=list[-1])
Removes and returns last object or obj from list

7   list.remove(obj)
Removes object obj from list

8   list.reverse()
Reverses objects of list in place

9   list.sort([func])
Sorts objects of list, use compare func if given



#LIST
append()    Adds an element at the end of the list
clear() Removes all the elements from the list
copy()  Returns a copy of the list
count() Returns the number of elements with the specified value
extend()    Add the elements of a list (or any iterable), to the end of the current list
index() Returns the index of the first element with the specified value
insert()    Adds an element at the specified position
pop()   Removes the element at the specified position
remove()    Removes the item with the specified value
reverse()   Reverses the order of the list
sort()  Sorts the list
```

In [15]:

```
str = "this is string example....wow!!!";
print ("str.capitalize() : ", str.capitalize())
str = "this is string example....wow!!!"
print ("str.center(40, 'a') : ", str.center(40, 'a'))
str = "this is string example....wow!!!";

sub = "i";
print ("str.count(sub, 4, 40) : ", str.count(sub, 4, 40))
sub = "wow";
print ("str.count(sub) : ", str.count(sub))
```

```
str.capitalize() :  This is string example....wow!!!
str.center(40, 'a') :  aaaathis is string example....wow!!!aaaa
```

In [20]:

```
#dictionary
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
del dict['Name']; # remove entry with key 'Name'
dict.clear();     # remove all entries in dict
del dict ;        # delete entire dictionary

print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
clear() Removes all the elements from the dictionary
copy()  Returns a copy of the dictionary
fromkeys()  Returns a dictionary with the specified keys and values
get()   Returns the value of the specified key
items() Returns a list containing the a tuple for each key value pair
keys()  Returns a list containing the dictionary's keys
pop()   Removes the element with the specified key
popitem()   Removes the last inserted key-value pair
setdefault()    Returns the value of the specified key. If the key does not exist: insert t
update()    Updates the dictionary with the specified key-value pairs
values()    Returns a list of all the values in the dictionary
```

```
  File "<ipython-input-20-5b885b69c109>", line 9
    clear()     Removes all the elements from the dictionary
                      ^
SyntaxError: invalid syntax
```

In [18]:

```
d = {'a': 1, 'b': 2}
print(d)
d['a'] = 100  # existing key, so overwrite
d['c'] = 3  # new key, so add
d['d'] = 4
print(d)
```

```
{'a': 1, 'b': 2}
{'a': 100, 'b': 2, 'c': 3, 'd': 4}
```

In [19]:

```
#array
cars = ["Ford", "Volvo", "BMW"]
cars.append("Honda")
cars.pop(1)
cars.remove("Volvo")
append()    Adds an element at the end of the list
clear() Removes all the elements from the list
copy()  Returns a copy of the list
count() Returns the number of elements with the specified value
extend()    Add the elements of a list (or any iterable), to the end of the current list
index() Returns the index of the first element with the specified value
insert()    Adds an element at the specified position
pop()   Removes the element at the specified position
remove()    Removes the first item with the specified value
reverse()   Reverses the order of the list
sort()  Sorts the list
```

```
  File "<ipython-input-19-d869dd2084b5>", line 6
    append()    Adds an element at the end of the list
                ^
SyntaxError: invalid syntax
```

In [ ]:

```

```