#Some important rules for Variables

#We can start with lowercase,uppercase,and

#even underscore but not with numbers,special

#case characters -->@,%,$....

'''a = 4512556

b = 35023565

print(a+b)

a1234 = 78956

age= 15

name = "code"

place = "vjwda"''' #Multi line comments -->Doc string

#built-in datatypes -->Numeric(int,float,complex)

#boolean(True(1)/False(0)),Collections -->Strings,Lists[],

#tuples(),sets,dictionaries...> {}

#Numeric -->int,float,complex

#Implicit type -->by default it understands type

a=5

#print(type(a))

b = 5.3

#print(type(b))

i3=2

#c = 5+3i

#print(c)

d = 5+3j

#print(d)

#print(type(d))

#Typecasting -->Converting one type to another

#Explicit Conversion

a =90

#int-->float,complex,bool

#print(type(a))

#Every built-in datatype -->built-in function

#b = float(a)

#print(b)

c = complex(a)

#print(c)

#Convert float -->complex,int

a=4.5

b = complex(a)

#print(b)

c = int(a)

#print(c)

#Convert complex-->int,float

a = 5+3j

#b = int(a)

#print(b)

#Complex cannot be converted to int/float

#boolean -->True/False

#bool(anything) -->True

#bool(nothing) -->False

c = bool(a)

#print(c)

d = bool(5.3)

#print(d)

e = bool(25)

#print(e)

#0,None,'',[].... -->boolean will be False

a = 5+2.3+(52+3j)+True

#print(a)

#Collections -->Strings(Group of Unicode characters)

#we enclose them in single/double quotes

a = "codegnan"

#print(a)

#print(type(a))

#len(obj) -->returns the number of objects/chrs in a collectn

#print(len(a))

#Indexing -->Starting postn is 0 and ends at len(obj)-1

#we use []

#print(a[0])

#print(a[6])

#print(a[56]) Indexerror -->as index is out of range

a = '12356' #digit string

#print(len(a))

#negative indexing -->from last -1 to len(obj)

b = "amreddy python codegnan"

#print(b[-1]) #last character

#print(len(b))

#print(b[22])

c = b[2]+b[-3]+b[5]+b[12] #'r'+'n'+'d'+'o' #String concatenation

#print(c)

#Slicing -->grp of charcters [start:end]

#start is included,end is excluded

#print(b[8:])

#print(b[8:14])

#print(b[12:45])

#print(b[45:]) #slicing returns empty character

b = "amreddy python codegnan"

#Slicing with same sign is always from lower index

#to higher index

#Strings are immutable

#print(b)

#print(b[1:5])

#print(b[5:0]) #not possible -->returns blank

#print(b[-5:-12])#empty

#print(b[-12:-5])

#let's go for mixed cases

#print(b[1:-3])

#print(b[-12:2])

a = "codegnan python"

#print(a[4:])

#a[4:] = "python" #Strings are immutable

#print(a)

#Striding -->[start:end:step]

#print(a[1:5:0]) #step cannot be zero

#print(a[1:5:1]) #a[1:5] -->answer

#print(a[1:5:2]) #a[1:5] -->'odeg' -->eliminate altrnte chr

#print(a[:8:3]) #a[0:8] -->'codegnan' -->cea (step-2)

#print(a[12:2:4])

#reverse order --> [::-1] it prints entire string in reverse order

#print(a)

#print(a[::-1])

#print(a[::-2]) #alternate character in reverse order

#Functions -->Case conversions,Joining,Splitting,

#Concatenation

a = "codeGNan"

#print(a)

#upper(),lower(),title(),capitalize()

#print(a.upper()) #converts to uppercase

b = a.upper()

#print(b)

c = a.lower()

#print(c)

#title() vs capitalize()

a = "codegnanpython"

#print(a.title()) #every frst lttr of words to upper

#print(a.capitalize())

#isupper(),islower(),isalpha(),isalnum(),isdigit()

#output returns boolean -->True/False

'''

print(a.isupper())

#isalpha() -->needs only alphabets

print(a.isalpha()) #space is also a character

print('code123'.isalpha())

print('code125'.isalnum()) #returns True for alphabets/

#numbers

print('12568'.isalnum())

print('1256'.isdigit()) #retrns True only for digitstring

'''

#count(),index(),find(),join(),split()

#count() -->returns the number of occurances in a string

a = "codegnan dgvfdamreddy"

'''print(a.count('d'))

#print(a.count('z')) #it not present -->0

#index() -->returns only first occurance

print(a.index('d'))

print(a.rindex('d')) #returns last occurance

print(a.index('d',3))

print(a.index('d',10))

print(a.index('d',14))

print(a.index('d',20)) #returns ValueError

a = "saketh code"

#print(a.index('q'))#returns ValueError

#find() -->overrides the error

print(a.find('e')) #returns frst occurance

print(a.find('q')) #returns -1

'''

#join(),split()

a="code";b="gnan"

c = a+b #string concatenation

#print(c)

d = a.join(b) #join(iterable) -->collection

#print(d)

e = "!".join("amreddy")

#print(e)

#split() -->divides according to given object (list)

d = a.split()

print(d)

print('code gnan java'.split()) #default split at space