In [4]:

```
x=5
b=28
c=x+b;
print("sum of c =",c)
```

```
sum of c = 33
```

In [9]:

```
a='saikumar'
b='ganji'
c=a+b
print(c)
```

```
saikumarganji
```

In [10]:

```
x=10;
print(type(x))
```

```
<class 'int'>
```

In [12]:

```
x='saikumar'
print(type(x))
```

```
<class 'str'>
```

In [16]:

```
x='a+j'
print(type(x))
```

```
<class 'str'>
```

In [18]:

```
a='saikumarganji'
print(a[0])
```

```
s
```

In [19]:

```
a='saikumarganji'
print(a[10])
```

```
n
```

In [24]:

```
for a in 'saikumarganji':
 print(a)
```

```
s
a
i
k
u
m
a
r
g
a
n
j
i
```

In [25]:

```
a='saikumarganji'
print(len(a))
```

```
13
```

In [26]:

```
txt = "The best things in life are free!"
if "free" in txt:
 print("Yes, 'free' is present.")
txt = "The best things in life are free!"
if "expensive" not in txt:
 print("Yes, 'expensive' is NOT present.")
```

```
Yes, 'free' is present.
Yes, 'expensive' is NOT present.
```

In [27]:

```
a='hello good morning'
if 'good' in a:
    print('its there')
b='where are you'
if 'the' not in b:
    print('not there')
```

```
its there
not there
```

In [29]:

```
b = "Hello, World!"
print(b[2:5])
# Negative Indexing
b = "Hello, World!"
print(b[-5:-2])
```

```
llo
orl
```

In [34]:

```
b = "asasihuahahsajhsjhnajsnaj!"
print(b[2:25])
# Negative Indexing
b = "saonasndiashacjhiaoshaaaaaaaaaaaaacio!"
print(b[-24:-2])
```

```
asihuahahsajhsjhnajsnaj
jhiaoshaaaaaaaaaaaaaci
```

In [35]:

```
a = "Hello, World!"
print(a.upper())
```

```
HELLO, WORLD!
```

In [36]:

```
a = "Hello, World!"
print(a.lower())
```

```
hello, world!
```

In [37]:

```
a = "Hello, World!"
print(a.strip())
```

```
Hello, World!
```

In [38]:

```
a = "Hello,bsbsWorld!"
print(a.strip())
```

```
Hello,bsbsWorld!
```

In [40]:

```
a = "Hello, World!"
print(a.replace("H", "J"))
```

```
Jello, World!
```

In [41]:

```
a='sai'
print(a.replace('s','ch'))
```

```
chai
```

In [42]:

```
a = "Hello, World!"
print(a.split())
```

```
['Hello,', 'World!']
```

In [43]:

```
thislist = ["apple", "banana", "cherry",'watermelon','guava']
print(thislist)
```

```
['apple', 'banana', 'cherry', 'watermelon', 'guava']
```

In [44]:

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

```
3
```

In [46]:

```
thislist = ["apple", "banana", "cherry",'watermelon','guava']
print(thislist[2])
```

```
cherry
```

In [47]:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[2:5])
```

```
['cherry', 'orange', 'kiwi']
```

In [48]:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[1:4])
```

```
['banana', 'cherry', 'orange']
```

In [49]:

```
thislist = ["apple", "banana", "cherry"]
if "apple" in thislist:
 print("Yes, 'apple' is in the fruits list")
```

```
Yes, 'apple' is in the fruits list
```

In [50]:

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(2, "watermelon")
print(thislist)
```

```
['apple', 'banana', 'watermelon', 'cherry']
```

In [58]:

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(2, "watermelon")
thislist.insert(4, "kiwi")
print(thislist)
```

```
['apple', 'banana', 'watermelon', 'cherry', 'kiwi']
```

In [54]:

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
print(thislist)
```

```
['apple', 'banana', 'cherry', 'orange']
```

In [57]:

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
thislist.append("mango")
thislist.append("pineapple")
print(thislist)
```

```
['apple', 'banana', 'cherry', 'orange', 'mango', 'pineapple']
```

In [59]:

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("banana")
print(thislist)
```

```
['apple', 'cherry']
```

In [60]:

```
thislist = ["apple", "banana", "cherry",2298]
thislist.remove("banana")
print(thislist)
```

```
['apple', 'cherry', 2298]
```

In [61]:

```
thislist = ["apple", "banana", "cherry",100101]
thislist.remove(100101)
print(thislist)
```

```
['apple', 'banana', 'cherry']
```

In [62]:

```
thislist = ["apple", "banana", "cherry"]
thislist.pop(1)
print(thislist)
```

```
['apple', 'cherry']
```

In [65]:

```
thislist = ["apple", "banana", "cherry",1,2,3,4,5]
thislist.pop(6)
print(thislist)
```

```
['apple', 'banana', 'cherry', 1, 2, 3, 5]
```

In [66]:

```
thislist = ["apple", "banana", "cherry"]
thislist.pop()
print(thislist)
```

```
['apple', 'banana']
```

In [68]:

```
thislist = ["apple", "banana", "cherry"]
del thislist
```

In [69]:

```
thislist = ["apple", "banana", "cherry"]
thislist.clear()
print(thislist)
```

```
[]
```

In [70]:

```
thislist = ["apple", "banana", "cherry"]
thislist.remove('apple')
print(thislist)
```

```
['banana', 'cherry']
```

In [71]:

```
thislist = ["apple", "banana", "cherry"]
for x in thislist:
 print(x)
```

```
apple
banana
cherry
```

In [72]:

```
for x in 'asndodjsojoaj':
    print(x)
```

```
a
s
n
d
o
d
j
s
o
j
o
a
j
```

In [73]:

```
thislist = ["apple", "banana", "cherry"]
for x in thislist:
 print(x)
```

```
apple
banana
cherry
```

In [75]:

```
for x in range(2, 30, 5):
 print(x)
```

```
2
7
12
17
22
27
```

In [79]:

```
# Store input numbers
num1 = input('Enter first number: ')
num2 = input('Enter second number: ')

# Add two numbers
sum = float(num1) + float(num2)

# Display the sum
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
```

```
Enter first number: 98392
Enter second number: 8298392
The sum of 98392 and 8298392 is 8396784.0
```

In [77]:

```
num1 = input('Enter first number: ')
num2 = input('Enter second number: ')

# Add two numbers
sum = float(num1) + float(num2)

# Display the sum
print('The sum of {0} and {1} is',sum)
```

```
Enter first number: 10
Enter second number: 29
The sum of {0} and {1} is 39.0
```

In [78]:

```
num1 = input('Enter first number: ')
num2 = input('Enter second number: ')

# Add two numbers
sum = float(num1) + float(num2)

# Display the sum
print('The sum =',sum)
```

```
Enter first number: 09
Enter second number: 87387
The sum = 87396.0
```

In [80]:

```
# Python program to swap two variables

x = 5
y = 10

# To take inputs from the user
#x = input('Enter value of x: ')
#y = input('Enter value of y: ')

# create a temporary variable and swap the values
temp = x
x = y
y = temp

print('The value of x after swapping: {}'.format(x))
print('The value of y after swapping: {}'.format(y))
```

```
The value of x after swapping: 10
The value of y after swapping: 5
```

In [81]:

```
a='sai'
b='kumar'
temp=a
a=b
b=temp
print('the value of a is',a)
print('the value of b is',b)
```

```
the value of a is kumar
the value of b is sai
```

In [85]:

```
a='sai'
b='kumar'
temp=a
a=b
b=temp
print('the value of a is:{}'.format(a))
print('the value of b is:{}'.format(b))
```

```
the value of a is:kumar
the value of b is:sai
```

In [88]:

```
a='sai'
b='kumar'
temp=a
a=b
b=temp
print('the value of a is:{}'.format(a))
print('the value of b is:{}'.format(b))
```

```
the value of a is:kumar
the value of b is:sai
```

In [89]:

```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

```
('apple', 'banana', 'cherry')
```

In [96]:

```
thistuple = ("apple", "banana", "cherry")
thistuple.insert("blackcurrant")

# the value is still the same:
print(thistuple)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
C:\Users\SAIKUM~1\AppData\Local\Temp/ipykernel_31104/4049799271.py in <modul
e>
      1 thistuple = ("apple", "banana", "cherry")
----> 2 thistuple.insert("blackcurrant")
      3
      4 # the value is still the same:
      5 print(thistuple)

AttributeError: 'tuple' object has no attribute 'insert'
```

In [98]:

```
thisset = {"apple", "banana", "cherry",'mango'}

thisset.update(["orange", "mango", "grapes"])

print(thisset)
```

```
{'orange', 'banana', 'cherry', 'grapes', 'mango', 'apple'}
```

In [102]:

```
thisset = {"apple", "banana", "cherry"}

thisset.update(["orange", "mango", "grapes"])

print(thisset)
```

```
{'orange', 'banana', 'cherry', 'grapes', 'mango', 'apple'}
```

In [100]:

```
thisdict ={
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

In [101]:

```
dict={'sai':'ganji','age':23,'city':'hyderabad'}
print(dict)
```

```
{'sai': 'ganji', 'age': 23, 'city': 'hyderabad'}
```

In [106]:

```
thisset = {"apple", "banana", "cherry"}

thisset.update(["orange", "mango", "grapes"])

print(thisset)
```

```
{'orange', 'banana', 'cherry', 'grapes', 'mango', 'apple'}
```

In [107]:

```
thisdict =  {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
for x, y in thisdict.items():
  print(x, y)
```

```
brand Ford
model Mustang
year 1964
```

In [110]:

```
thisdict ={
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
for x, y in thisdict.items():
  print(thisdict)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

In [111]:

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

```
a and b are equal
```

In [121]:

```
a='saikumar'
b='ganji'
if a>b:
 print("b is greater")
else:
 print('a is greater')
```

```
b is greater
```

In [122]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
  print(x)
```

```
apple
cherry
```

In [123]:

```
fruits = ["apple", "banana", "cherry",1,2,3,4,5,56,7,8,10]
for x in fruits:
  if x == "banana":
    continue
  print(x)
```

```
apple
cherry
1
2
3
4
5
56
7
8
10
```

In [124]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    break
  print(x)
```

```
apple
```

In [125]:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    pass
  print(x)
```

```
apple
banana
cherry
```

In [131]:

```
#alpha=[1,2,3,4,5,6,7,8,9,12,22,33,44,55]
for i in 'alpha':
 print(i)
```

```
a
l
p
h
a
```

In [142]:

```
alpha=[1,2,3,4,5,6,7,8,9,12,22,33,44,55]
for i in alpha:
 if i==8:
    break
 print(i)
```

```
1
2
3
4
5
6
7
```

In [143]:

```
alpha=[1,2,3,4,5,6,7,8,9,12,22,33,44,55]
for i in alpha:
 if i==8:
    continue
 print(i)
```

```
1
2
3
4
5
6
7
9
12
22
33
44
55
```

In [144]:

```
alpha=[1,2,3,4,5,6,7,8,9,12,22,33,44,55]
for i in alpha:
 if i==8:
    pass
 print(i)
```

```
1
2
3
4
5
6
7
8
9
12
22
33
44
55
```

In [145]:

```
def my_function(country = "Norway"):
  print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
```

```
I am from Sweden
I am from India
I am from Norway
I am from Brazil
```

In [146]:

```
def my_function(country = "Norway"):
  print("I am from " + country)

my_function()
my_function()
my_function()
my_function()
```

```
I am from Norway
I am from Norway
I am from Norway
I am from Norway
```

In [153]:

```
def my_function(country='australia'):
  print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function("usa")
my_function("Brazil")
#my_function()
```

```
I am from Sweden
I am from India
I am from usa
I am from Brazil
```

In [156]:

```
def my_function(country = "Norway"):
  print("I am from " + country)


my_function()
```

```
I am from Norway
```

In [160]:

```
def my_function():
  print("I am from ")


my_function()
```

```
I am from
```

In [161]:

```
def my_function(country='usa'):
  print("I am from " + country)


my_function()
```

```
I am from usa
```

In [162]:

```
x = lambda a, b, c: a + b + c
print(x(5, 6, 2))
```

```
13
```

In [164]:

```
cars = ["Ford", "Volvo", "BMW"]

for x in cars:
  print(x)
```

```
Ford
Volvo
BMW
```

In [165]:

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):
    print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
```

```
Hello my name is John
```

In [166]:

```
x = lambda a, b, c: a + b + c
print(x(7798,8028389283,2083028038))
```

```
10111425119
```

In [167]:

```
x = lambda a, b, c: a + b + c
print(x(5, 6, 2))
```

```
13
```

In [168]:

```
class Person:

    # init method or constructor
    def __init__(self, name):
        self.name = name

    # Sample Method
    def say_hi(self):
        print('Hello, my name is', self.name)

p = Person('Nikhil')
p.say_hi()
```

```
Hello, my name is Nikhil
```

In [169]:

```
for x in 'saikumar':
    print(x)
```

```
s
a
i
k
u
m
a
r
```

In [170]:

```
thislist=('apple','banana','cherry')
thislist.insert('guava')
print(thislist)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
C:\Users\SAIKUM~1\AppData\Local\Temp/ipykernel_31104/519796128.py in <module
>
      1 thislist=('apple','banana','cherry')
----> 2 thislist.insert('guava')
      3 print(thislist)

AttributeError: 'tuple' object has no attribute 'insert'
```

In [174]:

```
thislist=['apple','banana','cherry']
thislist.insert(2,'guava')
print(thislist)
```

```
['apple', 'banana', 'guava', 'cherry']
```

In [175]:

```
thislist=['apple','banana','cherry']
thislist.insert(3,'guava')
print(thislist)
```

```
['apple', 'banana', 'cherry', 'guava']
```

In [177]:

```
thislist=['apple','banana','cherry']
thislist.append('guava')
print(thislist)
```

```
['apple', 'banana', 'cherry', 'guava']
```

In [184]:

```
i = 1
while i < 6:
 print(i)
 i += 1
```

```
1
2
3
4
5
```

In [ ]:

```
i = 1
while i < 6:
 print(i)
```

In [186]:

```
i=1
while i<28:
  print(i)
  i += 1
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
```

In [185]:

```
i = 1
while i < 6:
 print(i)
 i += 1
```

```
1
2
3
4
5
```

In [192]:

```
for x in'banana':
 print(x)
```

```
b
a
n
a
n
a
```

In [193]:

```
def local_function():
 print("This is a local function")
local_function()
```

```
This is a local function
```

In [196]:

```
def func():
  print("hello sai")
func()
```

```
hello sai
```

In [197]:

```
def function():
 print('functions')
function()
```

```
functions
```

In [208]:

```
def outer_func():
    def inner_func():
      print("This is a nested function")
    inner_func()
outer_func()
```

```
This is a nested function
```

In [209]:

```
def my_function(fname):
  print(fname + " Refsnes")

my_function("Emil")
my_function("Tobias")
my_function("Linus")
```

```
Emil Refsnes
Tobias Refsnes
Linus Refsnes
```

In [210]:

```
def my_function(fname, lname):
  print(fname + " " + lname)

my_function("Emil", "Refsnes")
```

```
Emil Refsnes
```

In [211]:

```
def my_function(fname, lname):
  print(fname + lname)

my_function("Emil", "Refsnes")
```

```
EmilRefsnes
```

In [212]:

```
def my_function(fname, lname):
  print(fname + "and " + lname)

my_function("Emil", "Refsnes")
```

```
Emiland Refsnes
```

In [213]:

```
def my_function(fname, lname):
  print(fname + " and " + lname)

my_function("Emil", "Refsnes")
```

```
Emil and Refsnes
```

In [214]:

```
# Accessing tuple elements using indexing
my_tuple = ('p','e','r','m','i','t')

print(my_tuple[0])   # 'p'
print(my_tuple[5])   # 't'

# IndexError: list index out of range
# print(my_tuple[6])

# Index must be an integer
# TypeError: list indices must be integers, not float
# my_tuple[2.0]

# nested tuple
n_tuple = ("mouse", [8, 4, 6], (1, 2, 3))

# nested index
print(n_tuple[0][3])       # 's'
print(n_tuple[1][1])       # 4
```

```
p
t
s
4
```

In [216]:

```
n_tuple = ("mouse", [8, 4, 6], (1, 2, 3))

# nested index
print(n_tuple[0][3])
print(n_tuple[1][1])
print(n_tuple[2][2])
```

```
s
4
3
```

In [217]:

```
try:
 f = open('demo1.txt')
 if f.name == 'demo123.txt':
 raise Exception
except IOError as e:
 print('First!')
except Exception as e:
 print('Second')
else:
 print(f.read())
 f.close()
finally:
 print("Executing Finally...")
print('End of program')
```

```
  File "C:\Users\SAIKUM~1\AppData\Local\Temp/ipykernel_31104/868728768.py",
 line 4
    raise Exception
    ^
IndentationError: expected an indented block
```

In [223]:

```
try:

    f = open('demo1.txt')
    if f.name == 'demo123.txt':
     raise Exception
except IOError as e:
 print('First!')
except Exception as e:
 print('Second')
else:
 print(f.read())
 f.close()
finally:
  print("Executing Finally...")
print('End of program')
```

```
First!
Executing Finally...
End of program
```

In [222]:

```
try:
 f = open('demo1.txt')
 if f.name == 'demo123.txt':
 raise Exception
except IOError as e:
 print('First!')
except Exception as e:
 print('Second')
else:
 print(f.read())
 f.close()
finally:
   print("Executing Finally...")
                          #print('End of program')
#Because demo1.txt file exist in my user fo
```

```
  File "C:\Users\SAIKUM~1\AppData\Local\Temp/ipykernel_31104/784225452.py",
 line 4
    raise Exception
    ^
IndentationError: expected an indented block
```

In [224]:

```
try:
  print(x)
except:
  print("An exception occurred")
```

```
a
```

In [225]:

```
try:
  print("Hello")
except:
  print("Something went wrong")
else:
  print("Nothing went wrong")
```

```
Hello
Nothing went wrong
```

In [226]:

```
try:
  print("Hello")
except:
  print("Something went wrong")
```

```
Hello
```

In [228]:

```
try:
  print(v)
except:
  print("Something went wrong")
finally:
  print("The 'try except' is finished")
```

```
Something went wrong
The 'try except' is finished
```

In [229]:

```
try:
  f = open("demofile.txt")
  try:
    f.write("Lorum Ipsum")
  except:
    print("Something went wrong when writing to the file")
  finally:
    f.close()
except:
  print("Something went wrong when opening the file")
```

```
Something went wrong when opening the file
```

In [230]:

```
x = -1

if x < 0:
  raise Exception("Sorry, no numbers below zero")
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
C:\Users\SAIKUM~1\AppData\Local\Temp/ipykernel_31104/2072555483.py in <modul
e>
      2
      3 if x < 0:
----> 4   raise Exception("Sorry, no numbers below zero")

Exception: Sorry, no numbers below zero
```

In [231]:

```
x = "hello"

if not type(x) is int:
  raise TypeError("Only integers are allowed")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\SAIKUM~1\AppData\Local\Temp/ipykernel_31104/1233933522.py in <modul
e>
      2
      3 if not type(x) is int:
----> 4   raise TypeError("Only integers are allowed")

TypeError: Only integers are allowed
```

In [232]:

```
# initialize the amount variable
amount = 10000
# check that You are eligible to
if(amount>2999):
 print("You are eligible to purchase Dsa Self Paced")
```

```
You are eligible to purchase Dsa Self Paced
```

In [233]:

```
# initialize the amount variable
amount = 1
# check that You are eligible to
if(amount>2999):
 print("You are eligible to purchase Dsa Self Paced")
```

In [234]:

```
# initialize the amount variable
amount = 1000
# check that You are eligible to
if(amount<2999):
 print("You are eligible to purchase Dsa Self Paced")
```

```
You are eligible to purchase Dsa Self Paced
```

In [243]:

```
class ContextManager():
    def __init__(self):
 print('init method called')

 def __enter__(self): #wrks before the control enters with
 print('enter method called')
 return self

 def __exit__(self, exc_type, exc_value, exc_traceback): #to exit the file automatically wh
 print('exit method called')

with ContextManager() as manager:
 print('with statement block')
```

```
  File "<tokenize>", line 3
    print('init method called')
    ^
IndentationError: unindent does not match any outer indentation level
```

In [241]:

```
my_list = [1, 5, 4, 6, 8, 11, 3, 12]

new_list = list(map(lambda x: x * 2 , my_list))

print(new_list)
```

```
[2, 10, 8, 12, 16, 22, 6, 24]
```

In [ ]: