Question-1

Write a program that inputs an integer value (stop_number) from the user and prints the sum of all numbers from 0 to stop_number. You can assume the user will enter a valid value.

In [3]:

```
stop_number = int(input("Please enter a stop number: "))

result = 0
for i in range(0, stop_number):
    result = result + i
    print(result)
print(result)
```

```
Please enter a stop number: 9
0
1
3
6
10
15
21
28
36
36
```

Question-2

Extend your program to also input the start_number from the user. In this case, your program will add all numbers from start_number to stop_number. You can assume that the user will enter two valid values.

In [4]:

```
start_number = int(input("Please enter a start number: "))
stop_number = int(input("Please enter a stop number: "))

result = 0
for i in range(start_number, stop_number):
  result += i

print(result)
```

```
Please enter a start number: 3
Please enter a stop number: 9
33
```

Question-3

Extend your program to check if the start_number is an integer between 0 and 100. If so, your program will continue to ask for the second input and perform calculations; otherwise, it prints an error and stops execution.

In [6]:

```
start_number = int(input("Please enter a start number: "))

if 0 <= start_number <= 100:
  stop_number = int(input("Please enter a stop number: "))

  if start_number <= stop_number <= 100:

    result = 0
    for i in range(start_number, stop_number):
```

```
      result += i
    print(result)
  else:
    print("Invalid stop number")


else:
  print("Invalid start number")
```

```
Please enter a start number: 23
Please enter a stop number: 55
1232
```

Modules

They are tools that contain some functions and attributes that enable us to perform some functions easily. It provides the opportunity to use these functions it hosts in different files and programs over and over again. Thanks to the modules, Python has become a much more useful and easy-to-implement.

In order to use any module in Python, we first need to 'import' it. Importing means making functions and attributes within one module available from within another program.

import

In [7]:

```
import random as rnd

secret = rnd.randint(1,100)

check = False

#guess = int(input("Please enter you guess: "))

for x in range(5):
  guess = int(input("Please enter you guess: "))
  if guess == secret:
    print("Congrats!!")
    check = True
    break
  elif guess < secret:
    print("Please enter a greater number!!")
  else:
    print("Please enter a smaller number!!")

if not check:
  print("Looseer..The number was: ", secret)
```

```
Please enter you guess: 5
Please enter a greater number!!
Please enter you guess: 30
Please enter a smaller number!!
Please enter you guess: 23
Please enter a smaller number!!
Please enter you guess: 12
Please enter a smaller number!!
Please enter you guess: 7
Please enter a greater number!!
Looseer..The number was:  9
```

Dictionaries

- **A data type in Python.**
- **Dictionaries are indexed by keys and these keys can be a String and integer type.**
- **Shown as key:value pairs and keys are the unique values.**
- **{}**

In [10]:

```
d = {}

print(d)
type(d)
```

```
{}
```

Out[10]:

```
dict
```

In [11]:

```
d = {"python":1, "course":2}
print(d)
```

```
{'python': 1, 'course': 2}
```

In [12]:

```
d2 = {"machine":"learning", "artificial":"intelligence" }
d2
```

Out[12]:

```
{'machine': 'learning', 'artificial': 'intelligence'}
```

In [14]:

```
d2["artificial"]
```

Out[14]:

```
'intelligence'
```

In [16]:

```
d2["java"] = "programming"
d2
```

Out[16]:

```
{'machine': 'learning', 'artificial': 'intelligence', 'java': 'programming'}
```

In [19]:

```
d2["ruby"] = "programming"
d2
```

Out[19]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'programming'}
```

In [20]:

```
d2["ruby"] = "language"
d2
```

Out[20]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language'}
```

In [22]:

```
d
```

Out[22]:

Out[22]:

```
{'python': 1, 'course': 2}
```

In [23]:

```
d["course"]
```

Out[23]:

```
2
```

In [25]:

```
d2["deep"] = "learning"
d2
#print(d2)
```

Out[25]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language',
 'deep': 'learning'}
```

In [26]:

```
d
```

Out[26]:

```
{'python': 1, 'course': 2}
```

In [27]:

```
print(d.keys())
```

```
dict_keys(['python', 'course'])
```

In [28]:

```
d2.keys()
```

Out[28]:

```
dict_keys(['machine', 'artificial', 'java', 'ruby', 'deep'])
```

In [29]:

```
d2.values()
```

Out[29]:

```
dict_values(['learning', 'intelligence', 'programming', 'language', 'learning'])
```

In [43]:

```
d2.items()
```

Out[43]:

```
dict_items([('machine', 'learning'), ('artificial', 'intelligence'), ('java', 'programmin
g'), ('ruby', 'language'), ('deep', 'learning')])
```

In [40]:

```
for k in d2.keys():
  print(k)
```

```
machine
artificial
java
ruby
deep
```

In [41]:

```
for v in d.values():
  print(v)
```

```
1
2
```

In [33]:

```
for v in d2.values():
  print(v)
```

```
learning
intelligence
programming
language
learning
```

In [39]:

```
d2
```

Out[39]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language',
 'deep': 'learning'}
```

In [47]:

```
for k,v in d2.items():
  print("Key:", k, "Value:", v)
```

```
Key: machine Value: learning
Key: artificial Value: intelligence
Key: java Value: programming
Key: ruby Value: language
Key: deep Value: learning
```

In [48]:

```
for k,v in d.items():
  if v == 2:
    print(k)
```

```
course
```

In [49]:

```
d
```

Out[49]:

```
{'python': 1, 'course': 2}
```

In [51]:

```
d["a"] = [3,4,5]
d
```

Out[51]:

```
{'python': 1, 'course': 2, 'a': [3, 4, 5]}
```

In [52]:

```
d.pop("python")
d
```

Out[52]:

```
{'course': 2, 'a': [3, 4, 5]}
```

In [53]:

```
d.pop("python")
d
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-53-f8be978652e7> in <module>
----> 1 d.pop("python")
      2 d

KeyError: 'python'
```

In [55]:

```
d
```

Out[55]:

```
{'course': 2, 'a': [3, 4, 5]}
```

In [56]:

```
len(d)
```

Out[56]:

```
2
```

In [58]:

```
"a" in d
```

Out[58]:

```
True
```

In [59]:

```
"aa" in d
```

Out[59]:

```
False
```

In [60]:

```
d2
```

Out[60]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language',
 'deep': 'learning'}
```

In [66]:

```
d2.get("java")
```

Out[66]:

```
'programming'
```

In [1]:

```
d2.get("Global AI Hub")
```

```
---------------------------------------------------------------------------
```

```
NameError                                 Traceback (most recent call last)
<ipython-input-1-9ce9dbfa9305> in <module>
----> 1 d2.get("Global AI Hub")

NameError: name 'd2' is not defined
```

In [68]:

```
d2
```

Out[68]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language',
 'deep': 'learning'}
```

In [69]:

```
del d2["deep"]
d2
```

Out[69]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language'}
```

In [70]:

```
d2
```

Out[70]:

```
{'machine': 'learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language'}
```

In [71]:

```
d2['machine'] = "Deep Learning"
d2
```

Out[71]:

```
{'machine': 'Deep Learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language'}
```

In [73]:

```
d2['machine'] = "Deep Learning"
d2
```

Out[73]:

```
{'machine': 'Deep Learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language'}
```

In [75]:

```
d2_copy = d2.copy()
d2_copy
```

Out[75]:

```
{'machine': 'Deep Learning',
 'artificial': 'intelligence',
```

```
 'java': 'programming',
 'ruby': 'language'}
```

In [77]:

```
d2_copy['deep'] = 'AI'
d2_copy
```

Out[77]:

```
{'machine': 'Deep Learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language',
 'deep': 'AI'}
```

In [78]:

```
d2
```

Out[78]:

```
{'machine': 'Deep Learning',
 'artificial': 'intelligence',
 'java': 'programming',
 'ruby': 'language'}
```

In [83]:

```
d4 = {"human":2,
      "cat":4,
      "spider":8
     }

for i in d4:
  leg = d4[i]
  print("%s has %d legs " % (i,leg))
  print(str(i) + " has " + str(leg) +" legs.")
```

```
human has 2 legs
human has 2 legs.
cat has 4 legs
cat has 4 legs.
spider has 8 legs
spider has 8 legs.
```

In [84]:

```
d4 = {"human":2, "cat":4, "spider":8}

for i,leg in d4.items():
  print(str(i) + " has " + str(leg) +" legs.")
```

```
human has 2 legs.
cat has 4 legs.
spider has 8 legs.
```

In [95]:

```
districts = {"İstanbul":["Bostancı", "Beşiktaş", "Kadıköy"],
           "Ankara":["Çankaya", "Gölbaşı", "Kızılcahamam"],
           "İzmir":["Çeşme","Bornova","Foça"]}
```

In [86]:

```
districts["İstanbul"]
```

Out[86]:

```
['Bostancı', 'Beşiktaş', 'Kadıköy']
```

In [87]:

```
type(districts)
```

Out[87]:

```
dict
```

In [88]:

```
type(districts["Ankara"])
```

Out[88]:

```
list
```

In [96]:

```
districts["İzmir"][0][0]
```

Out[96]:

```
'Ç'
```

In [97]:

```
districts["İzmir"][2]
```

Out[97]:

```
'Foça'
```

In [98]:

```
#creating a dictionary from a list


nums = list(range(9)) # [0,1,2,3,4,5,6,7,8]

even_sqr = {x: x**2 for x in nums if x % 2 == 0}
print(even_sqr)
```

```
{0: 0, 2: 4, 4: 16, 6: 36, 8: 64}
```

Sets

- **A set, unlike lists, is a collection of data in no particular order; items cannot be accessed by indexing.**
- **Just like mathematical sets, it cannot contain more than one of the same items.**
- **An item can only be added once in sets. So a set cannot have two identical elements.**
- **We use the set() function to create an empty set.**

In [100]:

```
s = {"python", 5,6,8,5,6,"abc", "python","python","python","python","python","python","python","python","python"}
s
```

Out[100]:

```
{5, 6, 8, 'abc', 'python'}
```

In [101]:

```
empty = set()
type(empty)
```

Out[101]:

```
set
```

In [102]:

```
empty2 = {}
```

```
type(empty2)
```

Out[102]:

```
dict
```

In [105]:

```
s2 = set(["python", 5,6,8,5,6,"abc", "python"])
print(s2)
```

```
{'abc', 5, 6, 8, 'python'}
```

In [112]:

```
ne = set("pineapple")
print(ne)
ne
```

```
{'l', 'n', 'i', 'a', 'p', 'e'}
```

Out[112]:

```
{'a', 'e', 'i', 'l', 'n', 'p'}
```

In [118]:

```
#Bu hata neden alındı.
y = {"a","b",{'a':[1,2,3]},5,6}
print(y)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-118-055bcfaf741a> in <module>
      1 #Bu hata neden alındı.
----> 2 y = {"a","b",{'a':[1,2,3]},5,6}
      3 print(y)

TypeError: unhashable type: 'dict'
```

In [120]:

```
s2
```

Out[120]:

```
{5, 6, 8, 'abc', 'python'}
```

In [121]:

```
6 in s2
```

Out[121]:

```
True
```

In [122]:

```
9 in s2
```

Out[122]:

```
False
```

In [123]:

```
len(s2)
```

Out[123]:

```
5
```

In [124]:

```
s2.add("ai")
s2
```

Out[124]:

```
{5, 6, 8, 'abc', 'ai', 'python'}
```

In [126]:

```
s2.add("ai")
print(s2)
len(s2)
```

```
{'abc', 5, 6, 8, 'python', 'ai'}
```

Out[126]:

```
6
```

In [127]:

```
s2.remove("ai")
print(s2)
```

```
{'abc', 5, 6, 8, 'python'}
```

In [129]:

```
#karekök bulma
from math import sqrt

#import math

print({x for x in list(range(10))})

print({sqrt(x) for x in list(range(10))})
```

```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
{0.0, 1.0, 2.0, 1.7320508075688772, 1.4142135623730951, 2.23606797749979, 2.4494897427831
78, 2.6457513110645907, 2.8284271247461903, 3.0}
```

Tuples

- **It is an ordered data structure, it has an index value like lists, it can contain all data types.**
- **Unlike lists, they have a structure that cannot be changed. If we have more than one unchangeable values, we can collect them in a bunch.**

In [131]:

```
tupl = (1,2,3,4,5)
print(tupl)
print(type(tupl))
```

```
(1, 2, 3, 4, 5)
<class 'tuple'>
```

In [132]:

```
tuple2 = ()
type(tuple2)
```

Out[132]:

```
tuple
```

In [133]:

```
print(tupl[3])
```

```
4
```

In [134]:

```
tupl[-2]
```

Out[134]:

```
4
```

In [135]:

```
tupl[:4]
```

Out[135]:

```
(1, 2, 3, 4)
```

In [136]:

```
dm3 = ("asli", 5, 8, "september")
dm3.index("september")
```

Out[136]:

```
3
```

In [137]:

```
dm3.count(5)
```

Out[137]:

```
1
```

In [139]:

```
dm4 = ("apple","pear","strawberry")

dm4[0] = "cherry"

print(dm4)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-139-4f4bd157cad3> in <module>
      1 dm4 = ("apple","pear","strawberry")
      2
----> 3 dm4[0] = "cherry"
      4
      5 print(dm4)

TypeError: 'tuple' object does not support item assignment
```

In [140]:

```
dm4 = ("apple","pear","strawberry")

dm4[3] = "cherry"

print(dm4)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-140-2bcdca579179> in <module>
      1 dm4 = ("apple","pear","strawberry")
      2
----> 3 dm4[3] = "cherry"
      4
      5 print(dm4)

TypeError: 'tuple' object does not support item assignment
```

In [143]:

```
dm4
```

Out[143]:

```
('apple', 'pear', 'strawberry')
```

In [144]:

```
dm4.remove("pear")
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-144-fd596d7b946d> in <module>
----> 1 dm4.remove("pear")

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

In [145]:

```
# creating a dictionary in which tuples are keys.

a = {(x, x+1): x for x in range(10)}
print(a)
```

```
{(0, 1): 0, (1, 2): 1, (2, 3): 2, (3, 4): 3, (4, 5): 4, (5, 6): 5, (6, 7): 6, (7, 8): 7, (
8, 9): 8, (9, 10): 9}
```

In [146]:

```
s5 = (5,6)

print(type(s5))
```

```
<class 'tuple'>
```

In [148]:

```
a[s5]
```

Out[148]:

```
5
```

In [149]:

```
a[(3,4)]
```

Out[149]:

```
3
```

Question

In [151]:

```
vegetables = ['squash', 'pea', 'carrot', 'potato']
vegetables.sort()
```

In [152]:

```
vegetables
```

Out[152]:

```
['carrot', 'pea', 'potato', 'squash']
```

In [153]:

```
vegetables
```

Out[153]:

['carrot', 'pea', 'potato', 'squasn']

In [154]:

```
fruit = ['watermelon','pineapple', 'apple', 'banana', 'apricot']
sorted(fruit)
```

Out[154]:

```
['apple', 'apricot', 'banana', 'pineapple', 'watermelon']
```

In [155]:

```
fruit
```

Out[155]:

```
['watermelon', 'pineapple', 'apple', 'banana', 'apricot']
```