In [1]:

```
1.  Create a dictionary with 5 US states as keys and and their capitals as values.
2.  print all (key, value) pairs -- what method do you use.
3.  print only keys.
4.  print only values.
5.  length of the dictionary
6.  make a list with these keys as elements.
7.  make a list with these values as elements.
8.  Swap the keys and values in the dictionary -- keys should become values and values should become keys.
9.  create another dictionary in which keys are US states and values as the length of each corresponding key.
10. create another dictionary in which keys are US states and values as the reversed string of each corresponding key e
11. create another dictionary in which keys are US states and values as the sum of ascii values of all the elements witl
12. get the value at key number 3
13. What’s the sum of lengths of key1 and key2
14. What’s the sum of lengths of value1 and value2
15. What’s the sum of lengths of key1 and value3
```

```
  Cell In [1], line 1
    1.  Create a dictionary with 5 US states as keys and and their capitals as values.
        ^
SyntaxError: invalid syntax
```

In [18]:

```python
#1. Create a dictionary with 5 US states as keys and and their capitals as values.
def dictionar(x):
    US_dict={
        "Colorado" : "Denver",
         "Texas" : "Austin",
         "California" : "Sacramento",
         "New York" : "Albany",
         "Florida" : "Tallahassee"
    }

print(US_dict)


```

```
{'Colorado': 'Denver', 'Texas': 'Austin', 'California': 'Sacramento', 'New York': 'Albany', 'Florida': 'Talla
hassee'}
```

In [96]:

```python
for x, y in US_dict.items():
    print(x, y)
```

```
Colorado Denver
Texas Austin
California Sacramento
New York Albany
Florida Tallahassee
```

In [20]:

```python
#2. print all (key, value) pairs -- what method do you use.
x= US_dict.items()
print(x)
```

```
dict_items([('Colorado', 'Denver'), ('Texas', 'Austin'), ('California', 'Sacramento'), ('New York', 'Alban
y'), ('Florida', 'Tallahassee')])
```

In [21]:

```python
#3. print only keys.
US_dict.keys()
```

Out[21]:

```
dict_keys(['Colorado', 'Texas', 'California', 'New York', 'Florida'])
```

In [22]:

```python
#4. print only values.
US_dict.values()
```

Out[22]:

```
dict_values(['Denver', 'Austin', 'Sacramento', 'Albany', 'Tallahassee'])
```

In [25]:

```python
#5. length of the dictionary
print(len(US_dict))
```

```
5
```

In [44]:

```python
#6. make a list with these keys as elements.
print(list(US_dict.keys()))


```

```
['Colorado', 'Texas', 'California', 'New York', 'Florida']
```

In [45]:

```python
#7. make a list with these values as elements.
print(list(US_dict.values()))
```

```
['Denver', 'Austin', 'Sacramento', 'Albany', 'Tallahassee']
```

In [47]:

```python
#7. make a list with these keys as elements.
def my_dict(i):
    my_list=[]
    for x in i.keys():
        my_list.append(x)
    return my_list

my_dict(US_dict)

```

Out[47]:

```
['Colorado', 'Texas', 'California', 'New York', 'Florida']
```

In [48]:

```python
#7. make a list with these values as elements.
def my_dict(i):
    my_list=[]
    for x in i.values():
        my_list.append(x)
    return my_list

my_dict(US_dict)

```

Out[48]:

```
['Denver', 'Austin', 'Sacramento', 'Albany', 'Tallahassee']
```

In [97]:

```python
#8. Swap the keys and values in the dictionary -- keys should become values and values should become keys.


new_dict = dict([(value, key) for key, value in US_dict.items()])
for i in new_dict:
    print(i, " :  ", new_dict[i])

```

```
Denver  :   Colorado
Austin  :   Texas
Sacramento  :   California
Albany  :   New York
Tallahassee  :   Florida
```

In [98]:

```python
print(new_dict)
```

```
{'Denver': 'Colorado', 'Austin': 'Texas', 'Sacramento': 'California', 'Albany': 'New York', 'Tallahassee': 'F
lorida'}
```

In [63]:

```python
#9. create another dictionary in which keys are US states and values as the length of each corresponding key
#def new_dict(dictionary):
#print(dict())

x=US_dict.items()
print(x)
new_dict=dict([(key,len(key)) for key,value in US_dict.items()])
for i in new_dict:
    print(i,": " , new_dict[i])



```

```
dict_items([('Colorado', 'Denver'), ('Texas', 'Austin'), ('California', 'Sacramento'), ('New York', 'Alban
y'), ('Florida', 'Tallahassee')])
Colorado :  8
Texas :  5
California :  10
New York :  8
Florida :  7
```

In [64]:

```python
#10.    create another dictionary in which keys are US states and values as the reversed string of each corresponding ke
x=US_dict.items()
print(x)
new_dict=dict([(key,key[::-1]) for key,value in US_dict.items()])
for i in new_dict:
    print(i,": " , new_dict[i])
```

```
dict_items([('Colorado', 'Denver'), ('Texas', 'Austin'), ('California', 'Sacramento'), ('New York', 'Alban
y'), ('Florida', 'Tallahassee')])
Colorado :  odaroloC
Texas :  saxeT
California :  ainrofilaC
New York :  kroY weN
Florida :  adirolF
```

In [85]:

```python
#11.    create another dictionary in which keys are US states and values as the sum of ascii values of all the elements
Y=US_dict.items()
print(Y)
new_dict=dict([(key,sum(ord(x) for x in key)) for key,value in US_dict.items()])
for x in new_dict.keys():
    print(x,": " , new_dict[x])


```

```
dict_items([('Colorado', 'Denver'), ('Texas', 'Austin'), ('California', 'Sacramento'), ('New York', 'Alban
y'), ('Florida', 'Tallahassee')])
Colorado :  819
Texas :  517
California :  1016
New York :  751
Florida :  705
```

In [95]:

```python
#12.    get the value at key number 3

print(list(US_dict.keys())[2])
```

```
California
```

In [92]:

```
#13.    What’s the sum of lengths of key1 and key2
keys_list = list(US_dict.keys())
values_list = list(US_dict.values())

x = len(keys_list[0] + keys_list[1])
#x=len("Colorado")+len("Texas")
print(x)


```

```
13
```

In [93]:

```
#14.    What’s the sum of lengths of value1 and value2
x=len(values_list[0] + values_list[1])
print(x)
```

```
12
```

In [94]:

```
#15.    What’s the sum of lengths of key1 and value3
x=len(keys_list[0] + values_list[2])
print(x)

```

```
18
```

In [1]:

```
zip(keys,values)
Take 2 lists, one list containg string  and the other values , create a dict using str a skeys and values as values
Take 2 list of integers of len 5 and return the sum of the harmonic means of each pair of numbers with same index
 round the sum and find the character value ord=char
Take 3 integer lists of length 5 and use a zip command to create the 4th list such that each element o fthe list is equa
use this 4 lists and then use a zip command to create a 5th list such that each element of the list is equal to the geon
Take a str s=florida and split all the characters
for the given string florida i have to create a dict containf keys are characters of the str and values are ascii values
```

```
  Cell In [1], line 2
    Take 2 lists, one list containg string  and the other values , create a dict using str a skeys and values
as values
         ^
SyntaxError: invalid syntax
```

In [3]:

```
#1.Take 2 lists, one list containg string  and the other values , create a dict using str a skeys and values as valueske
values_list=[1,2,3,4,5]
result=dict(zip(keys_list,values_list))
print(result)
```

```
{'Anna': 1, 'John': 2, 'Rachel': 3, 'Ross': 4, 'Joey': 5}
```

In [87]:

```
#2.Take 2 list of integers of len 5 and return the sum of the harmonic means of each pair of numbers with same index
def harmonic_mean(list_x,list_y):
    result=[]
    for i,j in zip(list_x,list_y):
        temp= 2*i*j/(i+j)
        result.append(temp)
    return result


new_list=harmonic_mean([101,102,103,104,105],[106,107,108,109,110])
print(new_list)





```

```
[103.43961352657004, 104.4401913875598, 105.44075829383887, 106.44131455399061, 107.44186046511628]
```

In [88]:

```
#round the sum and find the character value ord=char
x=[]
for i in new_list:
    x.append(round(i))
print(x)
y= [chr(a) for a in x]
print(y)


```

```
[103, 104, 105, 106, 107]
['g', 'h', 'i', 'j', 'k']
```

In [119]:

```
#Take 3 integer lists of length 5 and use a zip command to create the 4th list such that each element o fthe list is equ
def arithmetic_mean(list_x, list_y, list_z):
    x=zip(list_x,list_y)

    result=[]
    #n=len()
    for i,j,k in zip(list_x,list_y,list_z):
        temp_mean=(i+j+k)/3
        result.append(temp_mean)
    return result

y=arithmetic_mean([1,2,3,4,5],[6,7,8,9,10], [11,12,13,14,15])
z=list(y)
print(z)
```

```
[6.0, 7.0, 8.0, 9.0, 10.0]
```

In [113]:

```
lis = [1,2,3,4]
print(lis)

def squares(i):
    return i*i

new_lis = list(map(squares, lis))
print(new_lis)
```

```
[1, 2, 3, 4]
[1, 4, 9, 16]
```

In [102]:

```
print(sum([1,2,3])/len([1,2,3]))
```

```
2.0
```

In [116]:

```
m = "Navya"
print(list(enumerate(m, start=2)))
```

```
[(2, 'N'), (3, 'a'), (4, 'v'), (5, 'y'), (6, 'a')]
```

In [126]:

```
 #use this 4 lists and then use a zip command to create a 5th list such that each element of the list is equal to the ge
def geometric_mean(list_x, list_y, list_z,list_a):


    result=[]

    for i,j,k,l in zip(list_x,list_y,list_z,list_a):
        temp_mean=(i*j*k*l)**(1/4)
        result.append(temp_mean)
    return result

y=geometric_mean([1,2,3,4,5],[6,7,8,9,10], [11,12,13,14,15],[6, 7, 8, 9, 10])
z=list(y)
print(z)
```

```
[4.460913442573438, 5.856010279957208, 7.068237686098791, 8.206694399204283, 9.306048591020996]
```

In [131]:

```
print(396**(1/4))
```

```
4.460913442573438
```

In [120]:

```
#Take a str s=florida and split all the characters
s="FLORIDA"
def split(word):
    return list(s)


s="FLORIDA"
print(split(s))
```

```
['F', 'L', 'O', 'R', 'I', 'D', 'A']
```

In [141]:

```
#for the given string florida create a dict containf keys are characters of the str and values are ascii values

s=['F', 'L', 'O', 'R', 'I', 'D', 'A']

def fun(x):
    return ord(x)

new_lis = list(map(fun, s))
print(new_lis)
new_list=dict(zip(s,new_lis))
print(new_list)

```

```
[70, 76, 79, 82, 73, 68, 65]
{'F': 70, 'L': 76, 'O': 79, 'R': 82, 'I': 73, 'D': 68, 'A': 65}
```

In [ ]:

```

```