```python
# Dictionary of dictionary

computers={
    'dell':{
        'make':'dell',
        'ram':'8gb',
        'processor_core':2,
        'hard_disk':'500gb'
    },
    'hp':{
        'make':'hp',
        'ram':'4gb',
        'processor_core':4,
        'hard_disk':'1tb'
    }
}

for k,v in computers.items():
    print(f"Make of computers ={v['make'].title()}")
    print(f"RAM of computer={v['ram']}")
    print(f"No of processor core={v['processor_core']}")
```

```
Make of computers =Dell
RAM of computer=8gb
No of processor core=2
Make of computers =Hp
RAM of computer=4gb
No of processor core=4
```

```python
# Task - 2

users={
    'aeinstein':{
        'first':'albert',
        'last':'einstein',
        'country':'germany'
    },
    'curie':{
        'first':'marie',
        'last':'curie',
        'country':'italy'
    }
}

for k,v in users.items():
    print(f"Full Name- {v['first'].title()} {v['last'].title()}")
    print(f"Location- {v['country'].title()}")
```

```
Full Name- Albert Einstein
Location- Germany
Full Name- Marie Curie
Location- Italy
```

collections

.lists

.tuples

.dictionary

.sets

```
# Sets
a set is unordered and unindexed


fruits={"apple","banana","cherry",'pineapple','orange','guava'}

for fruit in fruits:
    print(fruit)
```

```
apple
pineapple
guava
banana
orange
cherry
```

```
fruits={"apple","banana","cherry",'pineapple','orange','guava','apple'
}
print(type(fruits))   #using typemethod u can know what is typeof data
```

```
<class 'set'>
```

```
for fruit in fruits:
    print(fruit)
# note:- apple is not repeated in output
```

```
apple
pineapple
guava
banana
orange
cherry
```

```
# list in a dictionary
# Task3
```

```
pizza_farmfresh={
    'crust':'thin',
    'toppings':['capsicum','extra-cheese']
}

for k,v in pizza_farmfresh.items():
    if(len(v)==1):
        print(f"select the crust:{v[0]}")
    else:
        print(f"toppings on pizza are:")
        for topping in v:
            print(topping)


toppings on pizza are:
t
h
i
n
toppings on pizza are:
capsicum
extra-cheese

pizza_farmfresh={
    'crust':'thin',
    'toppings':['capsicum','extra-cheese']
}
print(f"topping name = {pizza_farmfresh['toppings'][0]}")

topping name = capsicum
```

Agenda

print() method

assign string to a variable

multiline string

looping through a struing

```
# print( ) method

# print() method
first_name='Albert'
last_name='Einstein'
print(first_name,last_name,sep='*')

Albert*Einstein
```

```
first_name='Albert'
last_name='Einstein'
print(first_name,last_name,sep='*',end='#')
print("hello how areyou")
```

```
Albert*Einstein#hello how areyou
```

```
first_name='Albert'
last_name='Einstein'
print(first_name,last_name,sep='*') # end ='/n'
print("hello how are you")
```

```
Albert*Einstein
hello how areyou
```

```
#assigning string to a variable

# multiline string
# Creating a multiline string
multiline_str = """I'm learning Python.
I refer to TechBeamers.com tutorials.
It is the most popular site for Python programmers."""
print("Multiline string: \n" + multiline_str)
```

```
Multiline string:
I'm learning Python.
I refer to TechBeamers.com tutorials.
It is the most popular site for Python programmers.
```

```
# looping thhrough string
# string are treated as array
message="do good find good"
for character in message:
    print(character)
```

```
d
o

g
o
o
d

f
i
n
d

g
o
o
d
```

```
message="do good find good"
print(message[3])
```

```
g
```

```
print(f"total no of characters={len(message)}")
```

```
total no of characters=17
```

```
# count() method

bikes=['honda','yamaha','suzuki','bajaj','bajaj']
number=bikes.count('bajaj')
print(number)
```

```
2
```

```
# find duplicate word in banana
list=[]
word= 'ankitnayan'
for character in word:
    list.append(character)
    number=list.count(character)
    #print(number)
    if(number>1):
        print(character)
# print(list)
```

```
n
a
a
n
```

```
# using set
word= 'banana'
# we are going to use set that does not allow duplicate value
word_converted_to_set=set(word)
print(word_converted_to_set)
```

```
{'b', 'a', 'n'}
```

```
# using count
word= 'mississippi'
for char in set(word):    # 0/p of set(word):- m,s,i,p
    count=word.count(char)
    if count>1:
        print(char,":",count)
```

```
s : 4
i : 4
p : 2

# len() function
# to get the length of string

# task 3
# check if string is palindrome or not
"""sos
racecar
never odd or even
12021"""
word='racecar'
length=len(word)
for i in range(length):
    if(word[i]==word[length-i-1]):
        print("True")
    else:
        print("false")

True
True
True
True
True
True
True

word='racecar'
word_to_list=list(word)
word_to_list.reverse()
if list(word)==word_to_list:
    print(f"{word} is a palindrome")
else:
    print(f"{word} is not a palinfdrome")

---------------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
<ipython-input-53-8d729b2370ca> in <module>
      1 word='racecar'
----> 2 word_to_list=list(word)
      3 word_to_list.reverse()
      4 if list(word)==word_to_list:
      5     print(f"{word} is a palindrome")

TypeError: 'list' object is not callable
```

```
message="Do good find good"
if 'good' in message:
    print("good is there in the message")
else:
    print("not there")

# Assignment
# Task 4
# there is a string
# 'fare: $18,369 - $120,379'  # string into integer
# write a logic to verify both the values are greater than zero
text = 'fare: $18,369 - $120,379'
number=int(text)
print(number)
```

```
---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
<ipython-input-54-511ba8e49837> in <module>
      4 # write a logic to verify both the values are greater than
zero
      5 text = 'fare: $18,369 - $120,379'
----> 6 number=int(text)
      7 print(number)

ValueError: invalid literal for int() with base 10: 'fare: $18,369 -
$120,379'
```

```
""remove , *,fare the u have to compare those numerical value
if it is greater than 0 print (fare is greater than 0)

be generic, i means your solutions not depend on change in some
value""

string="Fare: $18,369 - $120,379*"

import re
result=re.sub('[\W_]+','',string)
print(result)
```

```
Fare18369120379
```

```
import re
ini_string = "123abcjw:, .@! eiw"
print ("initial string : ", ini_string)
result = re.sub('[\W_]+', '', ini_string)

print ("final string", result)
```

```
initial string :  123abcjw:, .@! eiw
final string 123abcjweiw
```

```
import re

pattern = '[A-Za-z:*$,-]+'
string = 'Fare:$18,369 - $120,379*'


result = re.sub(pattern,'',string) # replacing pattern in place of
nothing('') in string

print('After replacement')
print(result)
```

```
After replacement
18369  120379
```

```
print(type(result))
len(result)
```

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

```
13
```

```
var1=""
var2=""
start=0
while result[start]!=" ":
    var1+=result[start];
    start+=1
print(start)
while start<=len(result)-1:
    var2+=result[start];
    start+=1
print(var1,var2)
```

```
5
18369   120379
```

```
fare1=int(var1)
fare2=int(var2)
print(fare1)
print(fare2)
```

```
18369
120379
```

```
print(type(fare1))
print(type(fare2))
```

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

```
if (fare1>0 and fare2>0):
    print("fare1 and fare2 is greater than zero")
```

```
fare1 and fare2 is greater than zero
```