```
#1

sub_marks = {'Phy': 93, 'Che': 94, 'Math': 91, 'Eng': 95, 'Bio': 90}
new = {}

for i,val in sorted(sub_marks.items()):
  new[i]=val

print(new)
```

```
{'Bio': 90, 'Che': 94, 'Eng': 95, 'Math': 91, 'Phy': 93}
```

```
#2

ub_marks = {'Phy': 93, 'Che': 94, 'Math': 91, 'Eng': 95, 'Bio': 90}

f = {}
for value in sorted(sub_marks.values()):
  for key in sub_marks.keys():
    if sub_marks[key] == value:
      f[key] = value
      break

print(f)
```

```
{'Bio': 90, 'Math': 91, 'Phy': 93, 'Che': 94, 'Eng': 95}
```

```
#3

sub_marks = {'Phy': 93, 'Che': 94, 'Math': 91, 'Eng': 95, 'Bio': 90}

keys = 0

values = 0

items = 0

for i in sub_marks:
  keys+=1
  values+=1
  items = keys + values

print(f'Total items inside the dictionary are {items}')
```

```
Total items inside the dictionary are 10
```

```
#3
sum_of_itmems = 0
```

```
sub_marks = {'Phy': 93, 'Che': 94, 'Math': 91, 'Eng': 95, 'Bio': 90}

for key,value in sub_marks.items():
  sum_of_itmems+=value

print(sum_of_itmems)
```

```
    463
```

```
#4

sub_marks = {'Phy': 93, 'Che': 94, 'Math': 91, 'Eng': 95, 'Bio': 90}

key_to_be_removed = 'Phy'

for key in sub_marks.keys():

  sub_marks.pop(key_to_be_removed)
  break

sub_marks
```

```
    {'Bio': 90, 'Che': 94, 'Eng': 95, 'Math': 91}
```

```
#5

sub_marks = {'Phy': 93, 'Che': 94, 'Math': 91, 'Eng': 95, 'Bio': 90}
marks = {'Sex Education':100}

for i in sub_marks:
  sub_marks.update(marks)
  break

sub_marks
```

```
    {'Bio': 90, 'Che': 94, 'Eng': 95, 'Math': 91, 'Phy': 93, 'Sex Education': 100}
```

```
#6

percentages = [50,90,87,45,65,20,36]

for i in percentages:
  if i >= 75:
    print(f'For {i} Grade is O')

  elif i<75 and i>=60:
    print(f'For {i} Grade is A')

  elif i<60 and i>=35:
    print(f'For {i} Grade is B')
    print(f'For {i} Grade is B')

  else:
    print('Fail')
```

```
For 50 Grade is B
For 90 Grade is O
For 87 Grade is O
For 45 Grade is B
For 65 Grade is A
Fail
For 36 Grade is B
```

```
#7

arr = ['cat', 'dog', 'tac', 'god', 'act']
```

```
list_alpha = list('abcdefghijklmnopqrstuvwxyz')

list_num = list(range(1,27))

alp_num_dict = {}

for i,key in enumerate(list_alpha):
  alp_num_dict[key]=list_num[i]

sum_list = []

for i in arr:
  sum = 0
  for j in i:
    sum += alp_num_dict[j]
  sum_list.append(sum)

zipped = dict(zip(arr,sum_list))

anagrams = {}

for value in sorted(zipped.values()):
  for key in zipped.keys():
    if zipped[key] == value:
      anagrams[key] = value

anagrams_list = list(anagrams.keys())
anagrams_list
```

```
['cat', 'tac', 'act', 'dog', 'god']
```

```
#8 Check if binary representations of two numbers are an anagram

num1 = 3
num2 = 2

bin1 = bin(num1)[2:]
```

```
bin2 = bin(num2)[2:]

zeros = abs(len(bin1)-len(bin2))

if len(bin1) > len(bin2):
  bin2 = bin2 + zeros*'0'

else:
  bin1 = bin1 + zeros*'0'

if (bin1.count('0') == bin2.count('0')) and (bin1.count('0') == bin1.count('0')):
  print('yes.binary representation of two numbers is an anagram')

else:
  print('no,binary representation of two numbers is\'nt an anagram')
```

```
no,binary representation of two numbers is'nt an anagram
```

```
#9

arr = ['cat', 'dog', 'tac', 'god', 'act']


list_alpha = list('abcdefghijklmnopqrstuvwxyz')

list_num = list(range(1,27))

alp_num_dict = {}

for i,key in enumerate(list_alpha):
  alp_num_dict[key]=list_num[i]

sum_list = []

for i in arr:
  sum = 0
  for j in i:
    sum += alp_num_dict[j]
  sum_list.append(sum)

subset = []
for i in sorted(sum_list):
  subset.append(sorted(sum_list).count(i))

print(f'Largest subset of anagrams is {max(subset)}')
```

```
Largest subset of anagrams is 3
```

```
#10

a = 5

input = 'datascience'
```

```
normal = 'abcdefghijklmnopqrstuvwxyz'
reverse = 'zyxwvutsrqponmlkjihgfedcba'

dic = dict(zip(normal,reverse))

pre = input[:a-1]

post = input[a-1:]

mirror = ''

for i in range(len(post)):
  mirror += dic[post[i]]

mirrored_string = pre+mirror

mirrored_string
```

```
'datahxrvmxv'
```

```
#11
a = ['a','b','c','a','b','k','d','l','j','h']

b = {}

for i in a:
  b[i]=a.count(i)

b
```

```
{'a': 2, 'b': 2, 'c': 1, 'd': 1, 'h': 1, 'j': 1, 'k': 1, 'l': 1}
```

```
#12

list_of_tuple = [('mumbai',10),('Delhi',20),('Kolkata',29)]

dict(list_of_tuple)
```

```
{'Delhi': 20, 'Kolkata': 29, 'mumbai': 10}
```

```
#13

collection = ['abc','zac','adr']

for word in collection:
    result = 'Word is ordered'
    i = 0
    l = len(word) - 1
    if (len(word) < 3):
      continue
    while i < l:
      if (ord(word[i]) > ord(word[i+1])):
        result = 'Word is not ordered'
        result = 'Word is not ordered'
        break
      else:
        i += 1
    if (result == 'Word is ordered'):
      print(word,': ',result)
```

```
abc :  Word is ordered
adr :  Word is ordered
```

```
#13

collection = ['abc','zac','adr','z']

for word in collection:
  if len(word)>=3:

    for j in range(len(word)-1):
      if ord(word[j])>ord(word[j+1]):
        result = 'not ordered'
        print(word,result)
        break

    else:
      result = 'ordered'
      print(word,result)
```

```
#14

a = [(i,i**3) for i in range(1,11)]

print(a)

new = dict(a)
print(new)
```

```
[(1, 1), (2, 8), (3, 27), (4, 64), (5, 125), (6, 216), (7, 343), (8, 512), (9, 729),
{1: 1, 2: 8, 3: 27, 4: 64, 5: 125, 6: 216, 7: 343, 8: 512, 9: 729, 10: 1000}
```

```
#15

input = [('Data', 24), ('Science', 8), ('Velocity', 30)]

dic = dict(input)

sorted_list_tuple = []

for value in sorted(dic.values()):
  for key in dic.keys():
    if dic[key] == value:
      sorted_list_tuple.append((key,value))

sorted_list_tuple
```

```
[('Science', 8), ('Data', 24), ('Velocity', 30)]
```

```
#16 insertion sort

def insertion_sort(lst):

  for i in range(1,len(lst)):
    curr_ele = lst[i]
    pos = i

    while curr_ele < lst[pos-1] and pos > 0:
      lst[pos] = lst[pos-1]
      pos = pos - 1

    lst[pos] = curr_ele

  return lst


insertion_sort([11,3,6,2,9])
```

```
[2, 3, 6, 9, 11]
```

```
#17 selection sort

def selection_sort(lst):

  for i in range(len(lst)):

    min_value = min(lst[i:])
    min_index = lst.index(min_value)
    lst[i],lst[min_index] = lst[min_index],lst[i]

  return lst

selection_sort([11,3,6,2,9])
```

```
[2, 3, 6, 9, 11]
```

```
# 18

def bubble_sort(l):

  for j in range(len(l)-1):
    for i in range(len(l)-1):
      if l[i]>l[i+1]:
        l[i],l[i+1] = l[i+1],l[i]

  return l

bubble_sort([11,3,6,2,9])
```

```
[2, 3, 6, 9, 11]
```

```
# 19

def merge_sort(lst):
  if len(lst)>1:
      mid = len(lst)//2
      left = lst[:mid]
      right = lst[mid:]
      merge_sort(left)
      merge_sort(right)
      i = 0
      j = 0
      k = 0

      while i < len(left) and j < len(right):
        if left[i] < right[j]:
          lst[k] = left[i]
          i = i+1
          k = k+1

        else:
          lst[k] = right[i]
          j = j+1
          k = k+1

      while i < len(left):
        lst[k] = left [i]
        i = i + 1
        k = k + 1

      while j < len(right):
        lst[k] = right [j]
        j = i + 1
        k = k + 1



  return lst


merge_sort([2,4,1,7,8,80])
```

```
[1, 2, 4, 7, 8, 80]
```

✓ 0s completed at 1:36 AM ● ×