In [1]:

```
# you are given a string your task is to count the frequency
# of letters of the string and print the letters in descending
# order of frequency
# input: aabbbccde
# output:
# b 3
# a 2
# c 2
# d 1
# e 1

from collections import Counter
str1 = 'aabbbccde'
result = [item for items, c in Counter(str1).most_common() for item in [items] * c]
str2 = ''.join([str(elem) for elem in result])
result.clear()
i = 0
while i != len(str2):
    temp = list()
    temp.append(str2[i])
    temp.append(str1.count(str2[i]))
    result.append(temp)
    i = i + str1.count(str2[i])
for i in range(len(result)):
    print(result[i][0], result[i][1])
```

```
b 3
a 2
c 2
d 1
e 1
```

In [2]:

```
# write a procedure to find min, max, mean, standard deviattion,
# variance of number list
# input: 10 50 80 70 49 23 11 4
# output: 4 80 37.13 27.25 848.70

import numpy as np
n = input()
l_n = n.split(' ')
l_n = [int(i) for i in l_n]
print(min(l_n))
print(max(l_n))
print(np.average(l_n))
print(np.std(l_n))
print(np.var(l_n))
```

```
10 50 80 70 49 23 11 4
4
80
37.125
27.25086007817001
742.609375
```

In [3]:

```
# you are given an integer array height of length n
# there are n vertical lines drawn such that the two
# endpoints of the ith line are (i, 0) and (i, height[i])
# find two lines that together with x axis form a container
# such that the container contains the most water
# return the maximum amount of water a container can store
# example 1
# input: height = [1, 8, 6, 2, 5, 4, 8, 3, 7]
# output: 49
#
# example 2
# input: height = [1, 1]
```

```
# output: 1

def maxArea(A, Len):
    area = 0
    for i in range(Len) :
        for j in range(i + 1, Len) :
            area = max(area, min(A[j], A[i]) * (j - i))
    return area

n = input()
l_n = n.split(' ')
l_n = [int(i) for i in l_n]
maxArea(l_n, len(l_n))
```

```
1 8 6 2 5 4 8 3 7
```

Out[3]:

```
49
```

In [4]:

```
# given a list of integers, write a program to print the count of
# all possible unique combinations of numbers whose sum is equal
# to K
# input: 2 4 6 1 3 (List of Integers)
#        6         (K)
# output: 3

from itertools import combinations
numbers = [int(n) for n in input().split()]
k = int(input())
count = 0
for i in range(1, len(numbers)+1):
    for c in combinations(numbers, i):
        if sum(c) == k:
            count += 1
print(count)
```

```
2 4 6 1 3
6
3
```

In [5]:

```
# Explain about the different types of exceptions in python
# with suitable example

# a = int(input())
# b = int(input())
# c = a/b
# print(c)
# ZeroDivisionError: division by zero
# NameError: name 'd' is not defined
# f = open('abc.txt', 'r')
# FileNotFoundError: [Errno 2] No such file or directory: 'abc.txt'
try:
    f = open('abc.txt', 'r')
    c = a/b
    print(c)
    print(d)
    f.close()
except ZeroDivisionError:
    print('Invalid Input: Divisor cannot be zero')
    pass # not helpful
except NameError:
    print('Non existing identifier passed')
except FileNotFoundError:
    print('File does not exists')
except:
    print('Something went wrong')
finally:
#     f.close() # NameError: name 'f' is not defined
    print('From finally block')
```

```
a = int(input())
if a < 18:
    raise Exception('You are underage, can not vote')
```

```
File does not exists
From finally block
10
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_12800/2217255966.py in <module>
     30 a = int(input())
     31 if a < 18:
---> 32     raise Exception('You are underage, can not vote')

Exception: You are underage, can not vote
```

In [6]:

```
# program to demonstrate the overriding of the base class method
# in the derived class

class Parent():
    def __init__(self):
        self.value = "Inside Parent"
    def show(self):
        print(self.value)
class Child(Parent):
    def __init__(self):
        self.value = "Inside Child"
    def show(self):
        print(self.value)
obj1 = Parent()
obj2 = Child()
obj1.show()
obj2.show()
```

```
Inside Parent
Inside Child
```