In [2]:

```
# digit frequency
l = [1,2,3,4,3,2,1,4,2]
m=[]
for i in l:
    if(i not in m):
        m.append(i)
c=dict.fromkeys(m,0)
print(c)
for i in l:
    a=l.count(i)
    c[i]=a
print(c)

```

```
{1: 0, 2: 0, 3: 0, 4: 0}
{1: 2, 2: 3, 3: 2, 4: 2}
```

In [1]:

```
#second largest number
li = list(map(int,input().split()))
#print(l)
li.sort()
print(li)
print(li[-2])
```

```
8 3 15 14 2
[2, 3, 8, 14, 15]
14
```

In [7]:

```
# digit frequency
#l = [1,2,3,4,3,2,1,4,2]
l = list(map(int,input().split()))
dic = {}
for i in l:
    if i in dic:
        dic[i] = dic[i]+1
    else:
        dic[i] = 1
print(dic)

```

```
1 2 3 4 3 2 1 4 2
{1: 2, 2: 3, 3: 2, 4: 2}
```

sets

-A set is an unordered collection data type that is itreable ,mutable and has no duplicate elements.

- python set is class represents the mathematical notation of set

In [8]:

```
s1 = set()
print(type(s1))
```

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

In [9]:

```
s2 = {1,2,3,4,5}
print(type(s2))
```

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

In [10]:

```
print(dir(set))
```

```
['__and__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc_
_', '__eq__', '__format__', '__ge__', '__getattribute__', '__gt__', '__hash_
_', '__iand__', '__init__', '__init_subclass__', '__ior__', '__isub__', '__i
ter__', '__ixor__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__o
r__', '__rand__', '__reduce__', '__reduce_ex__', '__repr__', '__ror__', '__r
sub__', '__rxor__', '__setattr__', '__sizeof__', '__str__', '__sub__', '__su
bclasshook__', '__xor__', 'add', 'clear', 'copy', 'difference', 'difference_
update', 'discard', 'intersection', 'intersection_update', 'isdisjoint', 'is
subset', 'issuperset', 'pop', 'remove', 'symmetric_difference', 'symmetric_d
ifference_update', 'union', 'update']
```

In [11]:

```
s1 = {1,2,3,4,5,9,10}
print(len(s1))
```

```
7
```

In [12]:

```
# add
s1.add(13)
print(s1)
```

```
{1, 2, 3, 4, 5, 9, 10, 13}
```

In [14]:

```
#update
s1.update([9.5,10,12])
print(s1)
```

```
{1, 2, 3, 4, 5, 9, 10, 9.5, 12, 13}
```

In [15]:

```
#copy
s2 =s1.copy()
print(s1)
print(s2)
```

```
{1, 2, 3, 4, 5, 9, 10, 9.5, 12, 13}
{1, 2, 3, 4, 5, 9, 10, 9.5, 12, 13}
```

In [16]:

```
# discard
s1.discard(10)
print(s1)

```

```
{1, 2, 3, 4, 5, 9, 9.5, 12, 13}
```

In [17]:

```
s1.discard(7)
```

In [18]:

```
s1
```

Out[18]:

```
{1, 2, 3, 4, 5, 9, 9.5, 12, 13}
```

In [19]:

```
# remove
s1.remove(9.5)
print(s1)
```

```
{1, 2, 3, 4, 5, 9, 12, 13}
```

In [20]:

```
s1.remove(0)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-20-3f824745aed8> in <module>
----> 1 s1.remove(0)

KeyError: 0
```

In [21]:

```
# pop
s1.pop()
print(s1)
```

```
{2, 3, 4, 5, 9, 12, 13}
```

In [23]:

```
s1.clear()
print(s1)
```

```
set()
```

```
## disjoint
- Two sets are said to be disjoint if they do not have any common elements
<img src=disjoint.jpg>
```

In [26]:

```
A = {1,2,3}
B = {5,7,9}
print(A.isdisjoint(B))
```

```
True
```

In [27]:

```
A = {1,2,7}
B = {5,7,9}
print(A.isdisjoint(B))
```

```
False
```

```
#### superset
-set A is said to the superset of B if all elements of B are in A
```

In [30]:

```
A={1,3,5,7,9}
B={1,3}
print(A.issuperset(B))
```

```
True
```

In [31]:

```
print(B.issubset(A))
```

```
True
```

In [32]:

```
#union
A = {1,3,5,6,7}
B = {1,2}
print(A.union(B))
```

```
{1, 2, 3, 5, 6, 7}
```

In [37]:

```
### intersection it return a set that contain the similarity between two or more sets.
A = {1,3,5,6,7}
B = {1,5,7}
res = print(A.intersection(B))
print(A)
```

```
{1, 5, 7}
{1, 3, 5, 6, 7}
```

In [39]:

```
#intersection_update
A = {1,3,5,6,7}
B = {1,5,7}
res = A.intersection_update(B)
print(res)
print(A)
print(B)
```

```
None
{1, 5, 7}
{1, 5, 7}
```

In [ ]:

```
## Difference
- The set difference of A and B is a set of element that exists only in set A not in B
<img src='difference.png'>
```

In [47]:

```
A = {'a','b','c','d'}
B = {'c','f','g'}
print(A-B)
print(B-A)
print(A.difference(B))
print(B.difference(A))
print(A)
```

```
{'d', 'a', 'b'}
{'g', 'f'}
{'d', 'a', 'b'}
{'g', 'f'}
{'d', 'c', 'a', 'b'}
```

In [46]:

```
#difference_update
A = {'a','b','c','d'}
B = {'c','f','g'}
print(A.difference_update(B))
#print(B.difference(A))
print(A)
```

```
None
{'d', 'a', 'b'}
```

In [ ]:

```
### symmetric _difference()
- the symmetric_difference of two sets A and B is the set of elements that are in eithe
But not in their intersection.
<img src='symmetric_difference.png'>
```

In [48]:

```
A={'a','b','c','d'}
B={'c','d','e'}
print(A.symmetric_difference(B))
print(A.symmetric_difference(B))
```

```
{'b', 'a', 'e'}
{'b', 'a', 'e'}
```

In [49]:

```
# symmetric_difference_update
A={'a','b','c','d'}
B={'c','d','e'}
print(A.symmetric_difference_update(B))
print(B.symmetric_difference_update(A))
print(A)
```

```
None
None
{'b', 'a', 'e'}
```

In [ ]:

```
<img src='apssdc.jpg'>
```

Sushikshanath Samriddhihi
Andhra Pradesh State
Skill
Development Corporation.

GET READY FOR THE SUMMER

In [ ]:

```

```