1.6 Comparision Operations for List.

In [4]:

```
x=["Dog","Cat","Rat"]
y=["Dog","Cat","Rat"]
z=["DOG","CAT","RAT"]
print(x==y)
print(x==z)
print(x!=z)
print(x>z)
print(x>=y)
print(x<=y)
x1=[50,20,30]
y1=[40,50,60,70,80,90]
print(x1>y1)
print(x1>=y1)
print(x1<y1)
print(x1<=y1)
```

```
True
False
True
True
True
True
True
True
False
False
```

1.7 Membership Operators for List.

1. in
2. not in

In [1]:

```
l=[10,20,30,40]
print(10 in l)
print(20 not in l)
```

```
True
False
```

1.8 Aliasing and Cloning of List.

In [2]:

```
l=[10,20,30,40]
x=l
print(x)
x[2]=100
print(x)
print(l)
```

```
[10, 20, 30, 40]
[10, 20, 100, 40]
[10, 20, 100, 40]
```

In [4]:

```
l=[10,20,30,40]
x=l[:]
x1=l.copy()
print(x)
print(x1)
x[2]=100
print(x)
print(l)
```

```
[10, 20, 30, 40]
[10, 20, 30, 40]
[10, 20, 100, 40]
[10, 20, 30, 40]
```

1.9 Nested List.

In [5]:

```
l=[10,20,30,[50,20,30],[30,60,50]]
print(len(l))
print(l[4])
print(l[4][2])
```

```
5
[30, 60, 50]
50
```

1.10 Nested List as Matrix.

In [14]:

```
l=[[10,20,30],[40,50,60],[70,80,90]]
for i in l:
    print(i)
for i in l:
    for j in i:
        print(j,end=" ")
    print()
```

```
[10, 20, 30]
[40, 50, 60]
[70, 80, 90]
10 20 30
40 50 60
70 80 90
```

1.11 List Comprehension.

1. syntax---> List=[expression for item in list if condition]

In [20]:

```
s=[x*x for x in range(1,11)]
print(s)
s1=[2**x for x in range(1,11)]
print(s1)
s2=[x for x in s if x%2==0]
print(s2)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
[2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]
[4, 16, 36, 64, 100]
```

In [27]:

```
#WAP to print the Largest even and Largest Odd numer in the list.
s=[x*x for x in range(1,11)]
even=0
odd=0
for i in s:
    if i%2==0:
        even=i
    else:
        odd=i
for i in s:
    if i%2==0:
        if i>odd:
            even=i
        else:
            if i>odd:
                odd=i
print(s)
print(even)
print(odd)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
100
81
```

In [28]:

```
def find_largest_even_odd(numbers):
    largest_even = None
    largest_odd = None

    for num in numbers:
        if num % 2 == 0:
            if largest_even is None or num > largest_even:
                largest_even = num
        else:
            if largest_odd is None or num > largest_odd:
                largest_odd = num

    return largest_even, largest_odd
```

```
# Example usage
numbers = [-45, -20, -80, -93, -3]
largest_even, largest_odd = find_largest_even_odd(numbers)
print(f"Largest even number: {largest_even}")
print(f"Largest odd number: {largest_odd}")
```

```
Largest even number: -20
Largest odd number: -3
```

2. Dictionary.

1. Reprsent a group of objedcts as key-value parts.
2. Duplicates keys are not allowed but values can be duplicated.
3. Heterogeneous objects are allowed for both keys and values.
4. Insertion order is not preserved.
5. Mutable.
6. Indexing and slicing concepts are not applicable.

In [33]:

```
d={}
print(type(d))
d[10]="Lucky"
d[20]='Arman'
d[30]='Aryan'
print(d)
del d[10]
print(d)
d.clear()
print(d)
```

```
<class 'dict'>
{10: 'Lucky', 20: 'Arman', 30: 'Aryan'}
{20: 'Arman', 30: 'Aryan'}
{}
```

2.1 Dictionary Comprenhension.

In [36]:

```
s={x:x*x for x in range(1,6)}
print(s)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

2.2 Important Functions of Dictonary.

1. dict()
2. len()
3. clear()
4. copy()
5. update()
6. get()
7. popitem()
8. keys()
9. values()
10. items()

In [40]:

```
s={x:x*x for x in range(1,6)}
x=s.copy()
x[1]=0
print(x)
print(s)
```

```
{1: 0, 2: 4, 3: 9, 4: 16, 5: 25}
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

In [42]:

```
d={1:'A',2:'B',3:'C'}
x={4:'C',5:'F',3:'E'}
d.update(x)
print(d)
```

```
{1: 'A', 2: 'B', 3: 'E', 4: 'C', 5: 'F'}
```

```
{1: 'A', 2: 'B', 3: 'E', 4: 'C', 5: 'F'}
```

In [45]:

```
d={1:'A',2:'B',3:'C'}
print(d[1])
print(d.get(1))
print(d.get(4))
print(d[4])
```

```
A
A
None
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-45-67322d52241d> in <module>
      3 print(d.get(1))
      4 print(d.get(4))
----> 5 print(d[4])

KeyError: 4
```

In [48]:

```
d={1:'A',2:'B',3:'C'}
print(d.get(1,"Guest"))
print(d.get(4,"Guest"))
```

```
A
Guest
```

In [49]:

```
d={1:'A',2:'B',3:'C'}
print(d.popitem())
print(d)
```

```
(3, 'C')
{1: 'A', 2: 'B'}
```

In [50]:

```
d={}
print(d.getitem())
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-50-842e49f566ef> in <module>
      1 d={}
----> 2 print(d.getitem())

AttributeError: 'dict' object has no attribute 'getitem'
```

In [52]:

```
d={1:'A',2:'B',3:'C'}
print(d.keys())
print(d.values())
print(d.items())
```

```
dict_keys([1, 2, 3])
dict_values(['A', 'B', 'C'])
dict_items([(1, 'A'), (2, 'B'), (3, 'C')])
```

In [ ]:

```
```