

## List Comprehension

Python provides a special syntax for creating list, set and dictionary called comprehension

Comprehension is applied only to mutable collections but not applicable to immutable collection

1. List comprehension
2. Set comprehension
3. Dictionary comprehension

List comprehension allows create list using special syntax

Syntax1: [value/expression for variable in iterable/iterator]

Syntax2: [value/expression for variable in iterable/iterator if test]

Comprehension allows including for loop and if within square brackets. It allows to create list in single including for loop and condition.

**Create list with Sqr of all the numbers from 1 to 10**

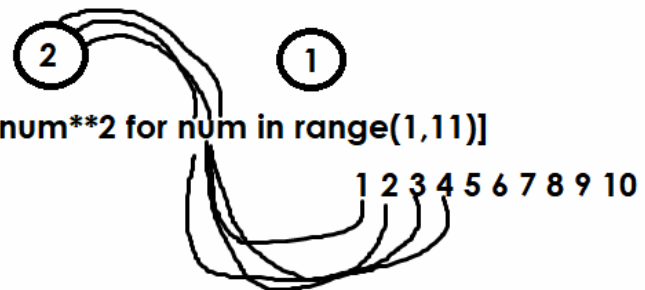
**without comprehension**

```
A=[]  
for num in range(1,11):  
    A.append(num**2)
```

```
print(A)
```

**with comprehension**

```
A=[num**2 for num in range(1,11)]
```



### Example:

# Without comprehension

```
A=[]  
for num in range(1,11):  
    A.append(num**2)
```

```
print(A)
```

# With comprehension

```
B=[num**2 for num in range(1,11)]
print(B)
```

### **Output**

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

### **Example:**

# Write a program to input n values in list

```
#without comprehension
n=int(input("How many values?"))
```

```
A=[]
for i in range(n):
    num=int(input("Enter Any number"))
    A.append(num)
```

```
print(A)
```

```
# with comprehension
B=[int(input("Enter Any number ")) for i in range(n)]
print(B)
```

### **Output**

```
How many values?5
Enter Any number10
Enter Any number20
Enter Any number30
Enter Any number40
Enter Any number50
[10, 20, 30, 40, 50]
Enter Any number 10
Enter Any number 20
Enter Any number 30
Enter Any number 40
Enter Any number 50
[10, 20, 30, 40, 50]
```

### **Example:**

# Write a program to read 2x2 matrix and display

#without comprehension

```
A=[]
for i in range(2):
    row=[]
    for j in range(2):
        value=int(input("Enter any value "))
        row.append(value)
    A.append(row)
```

```
print(A)
```

# with comprehension

```
B=[[int(input("Enter any value ")) for j in range(2)] for i in range(2)]
```

```
print(B)
```

### **Output**

Enter any value 1

Enter any value 2

Enter any value 3

Enter any value 4

```
[[1, 2], [3, 4]]
```

Enter any value 1

Enter any value 2

Enter any value 3

Enter any value 4

```
[[1, 2], [3, 4]]
```

### **Example:**

# Write a program to print sum of 2 matrices (2x2)

# using comprehension

```
print("Input Elements of first matrix")
```

```
A=[[int(input("enter value")) for j in range(2)] for i in range(2)]
```

```
print("Input Elements of second matrix")
```

```
B=[[int(input("Enter value")) for j in range(2)] for i in range(2)]
```

```
C=[[A[i][j]+B[i][j] for j in range(2)] for i in range(2)]
```

```
print(A,B,C,sep="\n")
```

### Output

Input Elements of first matrix

enter value1

enter value2

enter value3

enter value4

Input Elements of second matrix

Enter value5

Enter value6

Enter value7

Enter value8

[[1, 2], [3, 4]]

[[5, 6], [7, 8]]

[[6, 8], [10, 12]]

**Syntax2:** [value/expression for variable in iterable/iterator if test]

This syntax created list with values based on condition or test

If condition is True, it evaluate expression

If condition is False, it does not evaluate expression

### Example:

```
A=[1,5,8,2,6,7,9,12,11,24,22,56,87,98]
```

```
B=[value for value in A if value%2==0]
```

```
C=[value for value in A if value%2!=0]
```

```
print(A)
```

```
print(B)
```

```
print(C)
```

### Output

```
[1, 5, 8, 2, 6, 7, 9, 12, 11, 24, 22, 56, 87, 98]
```

```
[8, 2, 6, 12, 24, 22, 56, 98]
```

[1, 5, 7, 9, 11, 87]

**Example:**

```
grade_list=[['naresh','A'],  
            ['suresh','A'],  
            ['ramesh','B'],  
            ['kishore','C'],  
            ['kiran','B']]
```

```
print(grade_list)  
grade_listA=[stud for stud in grade_list if stud[1]=='A']  
grade_listB=[stud for stud in grade_list if stud[1]=='B']  
grade_listC=[stud for stud in grade_list if stud[1]=='C']
```

```
print(grade_listA)  
print(grade_listB)  
print(grade_listC)
```

**Output**

```
[['naresh', 'A'], ['suresh', 'A'], ['ramesh', 'B'], ['kishore', 'C'], ['kiran', 'B']]  
[['naresh', 'A'], ['suresh', 'A']]  
[['ramesh', 'B'], ['kiran', 'B']]  
[['kishore', 'C']]
```

**Example:**

```
students=[['naresh',70,80],  
          ['suresh',80,90],  
          ['ramesh',30,40],  
          ['kishore',80,60],  
          ['kiran',40,20]]
```

```
print(students)
```

```
students_result=[[stud[0],stud[1],stud[2],"pass" if stud[1]>=45 and  
                  stud[2]>=45 else "fail"]  
                 for stud in students]
```

```
print(students_result)
```

## Output

```
[[ 'naresh', 70, 80], [ 'suresh', 80, 90], [ 'ramesh', 30, 40], [ 'kishore', 80, 60],  
[ 'kiran', 40, 20]]  
[[ 'naresh', 70, 80, 'pass'], [ 'suresh', 80, 90, 'pass'], [ 'ramesh', 30, 40, 'fail'],  
[ 'kishore', 80, 60, 'pass'], [ 'kiran', 40, 20, 'fail']]
```

## Copy of the list or clone

How to create copy of the list?

Python allows creating copy of list in 2 ways (OR) there are two types of copies support by python

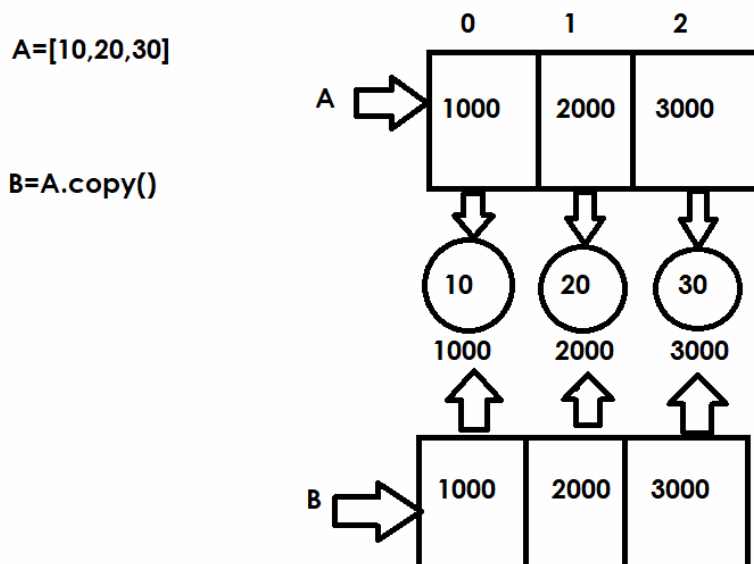
1. Shallow Copy
2. Deep Copy

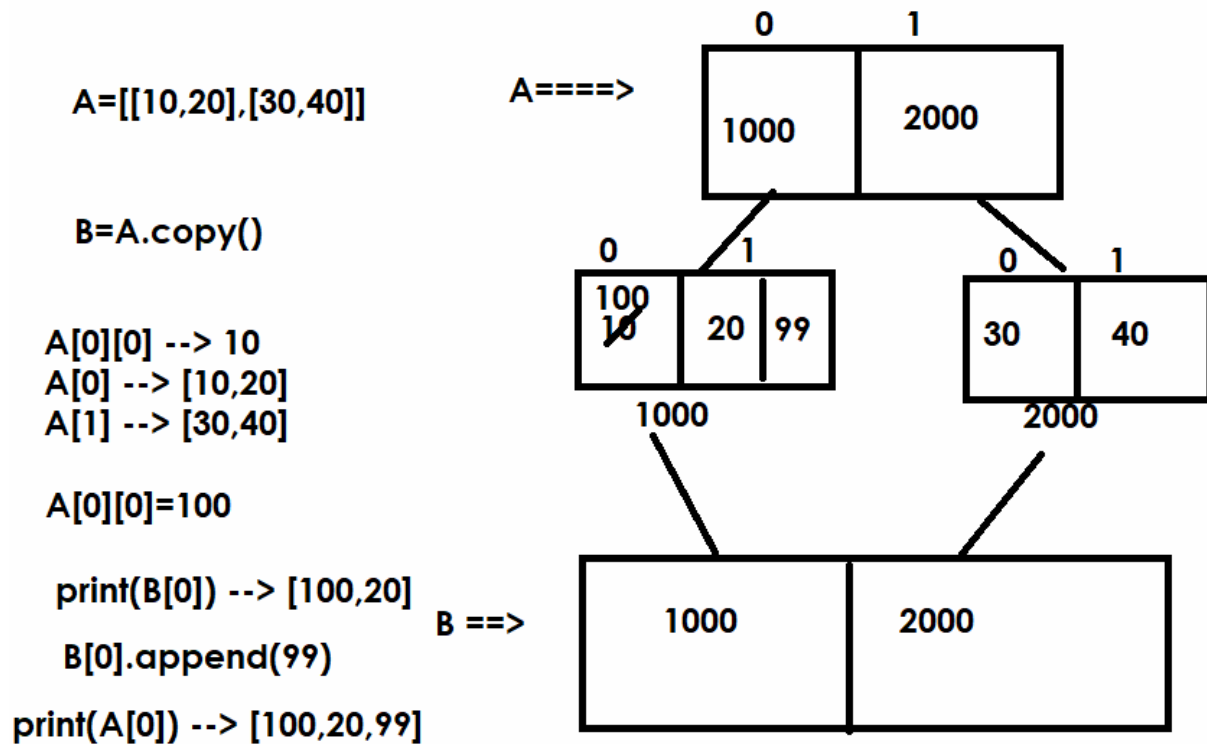
### Shallow Copy

copy() method of list create shallow copy

`<variable-name>=<list-name>.copy()`

Shallow copy create new list by copying address/reference of objects found existing list.





### Example:

# creating copy with immutable objects

`A=[10,20,30]`

`B=A.copy()`

`print(A)`

`print(B)`

`A[0]=99`

`print(A)`

`print(B)`

# creating copy with mutable objects

`X=[[10,20],[30,40]]`

`Y=X.copy()`

`print(X)`

`print(Y)`

`X[0][0]=99`

`print(X)`

`print(Y)`

`Y[0].append(88)`

`print(X)`

`print(Y)`

## Output

```
[10, 20, 30]
[10, 20, 30]
[99, 20, 30]
[10, 20, 30]
[[10, 20], [30, 40]]
[[10, 20], [30, 40]]
[[99, 20], [30, 40]]
[[99, 20], [30, 40]]
[[99, 20, 88], [30, 40]]
[[99, 20, 88], [30, 40]]
```

## Deep Copy

Deep copy is object copy, in deep copy a new list created by copying objects found in existing list.

This deep copy is implemented using “**copy**” module. It is standard module which comes with python software

This module provides a predefined function called **deepcopy()**

## Syntax:

Import copy

<variable-name>=copy.deepcopy(list-name)