

Lab2-1:

```
Array_A = [5,7,9,11,13]
```

```
SumArray_A = 0
```

```
print(("ArrayA= ")+str(Array_A))
```

```
Array_Length = len(Array_A)
```

```
print(("Array Lenght = ")+str(Array_Length))
```

```
for x in range(Array_Length):
```

```
    SumArray_A += Array_A[x]
```

```
print(("Sumation of Array = ")+str(SumArray_A))
```

**Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java**

Python 3.6  
(known limitations)

```
1 Array_A = [5,7,9,11,13]
2 SumArray_A = 0
3
4 print(("ArrayA= ")+str(Array_A))
5 Array_Length = len(Array_A)
6
7 print(("Array Lenght = ")+str(Array_Length))
8
9 for x in range(Array_Length):
10
11     SumArray_A += Array_A[x]
12
13 print(("Sumation of Array = ")+str(SumArray_A))
```

[Edit this code](#)

→ line that just executed  
→ next line to execute

Print output (drag lower right corner to resize)

```
ArrayA= [5, 7, 9, 11, 13]
Array Lenght = 5
Sumation of Array = 45
```

Frames

Global frame
Array_A
SumArray_A 45
Array_Length 5
x 4

Objects

list
0 5
1 7
2 9
3 11
4 13

Done running (17 steps)

[Customize visualization](#)

Lab2-2:

```
Array_A = [7,5,10,14,3,9,7]
```

```
Array_B = [9,10,3,4,2,5,7,1]
```

```
Array_C = []
```

```
ValueA = 0
```

```
ValueB = 0
```

```
ValueC = 0
```

```
print(("ArrayA = ")+str(Array_A))
```

```
print(("ArrayB = ")+str(Array_B))
```

```
Array_LengthA = len(Array_A)
```

```
Array_LengthB = len(Array_B)
```

```
print(("Array LengthA = ")+str(Array_LengthA))
```

```
print(("Array LengthB = ")+str(Array_LengthB))
```

```
ValueA = Array_A.count(7)
```

```
ValueB = Array_B.count(7)
```

```
print(("7 in ArrayA = ")+str(ValueA))
```

```
print(("7 in ArrayB = ")+str(ValueA))
```

```
Array_A.append(1)
```

```
Array_B.append(14)
```

```
Array_C = Array_A.copy()
```

```
Array_C.extend(Array_B)
```

```
ValueC = Array_C.count(7)
```

```
Array_C.sort()
```

```
Array_C.remove(7)
```

```
Array_D = Array_C.copy()
```

```
Array_D.reverse()
```

```
print("ArrayC = " + str(Array_C))
```

```
print("ArrayD = " + str(Array_D))
```

**Python Tutor: Visualize code in Python, JavaScript, C, C++ , and Java**

Python 3.6  
(known limitations)

```
1 Array_A = [7,5,10,14,3,9,7]
2 Array_B = [9,10,3,4,2,5,7,1]
3 Array_C = []
4 ValueA = 0
5 ValueB = 0
6 ValueC = 0
7
8 print(("ArrayA = ") + str(Array_A))
9 print(("ArrayB = ") + str(Array_B))
10
11 Array_LengthA = len(Array_A)
12 Array_LengthB = len(Array_B)
13 print(("Array LengthA = ") + str(Array_LengthA))
14 print(("Array LengthB = ") + str(Array_LengthB))
15
16 ValueA = Array_A.count(7)
17 ValueB = Array_B.count(7)
18
19 print(("7 in ArrayA = ") + str(ValueA))
20 print(("7 in ArrayB = ") + str(ValueA))
```

Print output (drag lower right corner to resize)

```
ArrayA = [7, 5, 10, 14, 3, 9, 7]
ArrayB = [9, 10, 3, 4, 2, 5, 7, 1]
Array LengthA = 7
Array LengthB = 8
7 in ArrayA = 2
7 in ArrayB = 2
ArrayC = [1, 1, 2, 3, 3, 4, 5, 5, 7, 7, 9, 9, 10, 10, 14, 14]
ArrayD = [14, 14, 10, 10, 9, 9, 7, 7, 5, 5, 4, 3, 3, 2, 1, 1]
```

Frames

Global frame

- Array\_A
- Array\_B
- Array\_C
- ValueA
- ValueB
- ValueC
- Array\_LengthA
- Array\_LengthB
- Array\_D

Objects

list

0	1	2	3	4	5	6	7
7	5	10	14	3	9	7	1

list

0	1	2	3	4	5	6	7	8
9	10	3	4	2	5	7	1	14

list

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	2	3	3	4	5	5	7	7	9	9	10	10	14	14

list

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
14	14	10	10	9	9	7	7	5	5	4	3	3	2	1	1

Done running (27 steps)

Customize visualization

Lab2-3:

```
Array_A= ["Number ID","Name","Count"]
```

```
Array_LenghtA = 0
```

```
Location_of_Name = 0
```

```
Array_LengthA = len(Array_A)
```

```
Location_of_Name = Array_A.index("Name")
```

```
Array_A.append("Status")
```

**Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java**

Python 3.6  
(known limitations)

```
1 Array_A= ["Number ID","Name","Count"]
2 Array_LenghtA = 0
3 Location_of_Name = 0
4
5 Array_LengthA = len(Array_A)
6 Location_of_Name = Array_A.index("Name")
7
8 Array_A.append("Status")
```

[Edit this code](#)

→ line that just executed  
→ next line to execute

Done running (6 steps)

Frames

Global frame
Array_A
Array_LenghtA
Location_of_Name
Array_LengthA

Objects

list
0
"Number ID"
1
"Name"
2
"Count"
3
"Status"

Customize visualization

## Lab2-4:

```
Array_A= [{"Number ID","Name","Count","Status"},]
```

```
Array_LenghtA = 0
```

```
Array_LengthA = len(Array_A)
```

```
Array_A.insert(1,["1","Rubber","0","Out of stock"])
```

```
Array_A.insert(2,["2","Ruler","0","In stock"])
```

```
Array_A.insert(3,["3","Pencil","0","In stock"])
```

**Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java**

Python 3.6  
(known limitations)

```
1 Array_A= [{"Number ID","Name","Count","Status"},]  
2 Array_LenghtA = 0  
3  
4 Array_LengthA = len(Array_A)  
5  
6 Array_A.insert(1,["1","Rubber","0","Out of stock"])  
7 Array_A.insert(2,["2","Ruler","0","In stock"])  
8 Array_A.insert(3,["3","Pencil","0","In stock"])
```

[Edit this code](#)

⇒ line that just executed  
→ next line to execute

Done running (6 steps)

Customize visualization

Frames

Global frame

Array_A	list
Array_LenghtA	0
Array_LengthA	1

Objects

list

0	1	2	3
"Number ID"	"Name"	"Count"	"Status"

list

0	1	2	3
"3"	"Pencil"	"0"	"In stock"

list

0	1	2	3
"1"	"Rubber"	"0"	"Out of stock"

list

0	1	2	3
"2"	"Ruler"	"0"	"In stock"

Lab2-5:

```
Array_A= [{"Number ID","Name","Count","Status"},]
```

```
Array_A.insert(1,["1","Rubber","0","Out of stock"])
```

```
Array_A.insert(2,["2","Ruler","5","In stock"])
```

```
Array_A.insert(3,["3","Pencil","1","In stock"])
```

```
Array_A.insert(4,["4","Pen","10","In stock"])
```

```
Array_A.insert(5,["5","Colour Pencil","5","In stock"])
```

```
Array_A.insert(6,["6","A4 Paper","0","Out of stock"])
```

```
print("-----In Stock-----")
```

```
for x in Array_A:
```

```
    if x[3] == "In stock":
```

```
        print(x)
```

```
print("-----Out of Stock-----")
```

```
for x in Array_A:
```

```
    if x[3] == "Out of stock":
```

```
        print(x)
```

```
#("-----Cut Stock-----")
```

```
Array_A[2][2] = 4
```

```
Array_A[3][2] = 0
```

```
Array_A[4][2] = 8
```

```
Array_A[5][2] = 4
```

```
Array_A[3][3] = "Out of stock"
```

```
print("-----Stock-----")
```

```
for x in Array_A:
```

```
    print(x)
```

## Python Tutor: Visualize code in Python, JavaScript, C, C++, and Java

Python 3.6  
(known limitations)

```
12 for x in Array_A:
13     if x[3] == "In stock":
14         print(x)
15 print("-----Out of Stock-----")
16 for x in Array_A:
17     if x[3] == "Out of stock":
18         print(x)
19
20
21 #("-----Cut Stock-----")
22 Array_A[2][2] = 4
23 Array_A[3][2] = 0
24 Array_A[4][2] = 8
25 Array_A[5][2] = 4
26
27 Array_A[3][3] = "Out of stock"
28
29 print("-----Stock-----")
30 for x in Array_A:
31     print(x)
```

Print output (drag lower right corner to resize)

```
-----In Stock-----
['2', 'Ruler', '5', 'In stock']
['3', 'Pencil', '1', 'In stock']
['4', 'Pen', '8', 'In stock']
['5', 'Colour Pencil', '5', 'In stock']
-----Out of Stock-----
['1', 'Rubber', '0', 'Out of stock']
['6', 'A4 Paper', '0', 'Out of stock']
-----Stock-----
['Number ID', 'Name', 'Count', 'Status']
['1', 'Rubber', '0', 'Out of stock']
['2', 'Ruler', 4, 'In stock']
['3', 'Pencil', 0, 'Out of stock']
['4', 'Pen', 8, 'In stock']
['5', 'Colour Pencil', 4, 'In stock']
['6', 'A4 Paper', '0', 'Out of stock']
```

Frames

Global frame

Array\_A

x

Objects

list

0	1	2	3
"Number ID"	"Name"	"Count"	"Status"

list

0	1	2	3
"3"	"Pencil"	0	"Out of stock"

list

0	1	2	3
"1"	"Rubber"	"0"	"Out of stock"

list

0	1	2	3
"2"	"Ruler"	4	"In stock"

list

0	1	2	3
"4"	"Pen"	8	"In stock"

list

0	1	2	3
"5"	"Colour Pencil"	4	"In stock"

list

0	1	2	3
"6"	"A4 Paper"	"0"	"Out of stock"

Done running (66 steps)

Customize visualization