

Python - 2-D Array

Two dimensional array is an array within an array. It is an array of arrays. In this type of array the position of an data element is referred by two indices instead of one. So it represents a table with rows and columns of data.

In the below example of a two dimensional array, observe that each array element itself is also an array.

Consider the example of recording temperatures 4 times a day, every day. Some times the recording instrument may be faulty and we fail to record data. Such data for 4 days can be presented as a two dimensional array as below.

```
Day 1 - 11 12 5 2
Day 2 - 15 6 10
Day 3 - 10 8 12 5
Day 4 - 12 15 8 6
```

The above data can be represented as a two dimensional array as below.

```
T = [[11, 12, 5, 2], [15, 6, 10], [10, 8, 12, 5], [12, 15, 8, 6]]
```

Accessing Values

The data elements in two dimensional arrays can be accessed using two indices. One index referring to the main or parent array and another index referring to the position of the data element in the inner array. If we mention only one index then the entire inner array is printed for that index position.

Example

The example below illustrates how it works.

```
from array import *  
  
T = [[11, 12, 5, 2], [15, 6, 10], [10, 8, 12, 5], [12, 15, 8, 6]]  
  
print(T[0])  
  
print(T[1][2])
```

Output

When the above code is executed, it produces the following result –

```
[11, 12, 5, 2]  
10
```

To print out the entire two dimensional array we can use python for loop as shown below. We use end of line to print out the values in different rows.

Example

```
from array import *
```

```
T = [[11, 12, 5, 2], [15, 6,10], [10, 8, 12, 5], [12,15,8,6]]
for r in T:
    for c in r:
        print(c,end = " ")
    print()
```

Output

When the above code is executed, it produces the following result –

```
11 12  5 2
15  6 10
10  8 12 5
12 15  8 6
```

Inserting Values

We can insert new data elements at specific position by using the insert() method and specifying the index.

Example

In the below example a new data element is inserted at index position 2.

```
from array import *
T = [[11, 12, 5, 2], [15, 6,10], [10, 8, 12, 5], [12,15,8,6]]

T.insert(2, [0,5,11,13,6])

for r in T:
    for c in r:
```

```
    print(c,end = " ")
print()
```

Output

When the above code is executed, it produces the following result –

```
11 12  5  2
15  6 10
 0  5 11 13 6
10  8 12  5
12 15  8  6
```

Updating Values

We can update the entire inner array or some specific data elements of the inner array by reassigning the values using the array index.

Example

```
from array import *

T = [[11, 12, 5, 2], [15, 6,10], [10, 8, 12, 5], [12,15,8,6]]

T[2] = [11,9]
T[0][3] = 7
for r in T:
    for c in r:
        print(c,end = " ")
    print()
```

Output

When the above code is executed, it produces the following result –

```
11 12 5 7
15 6 10
11 9
12 15 8 6
```

Deleting the Values

We can delete the entire inner array or some specific data elements of the inner array by reassigning the values using the `del()` method with index. But in case you need to remove specific data elements in one of the inner arrays, then use the update process described above.

Example

```
from array import *
T = [[11, 12, 5, 2], [15, 6,10], [10, 8, 12, 5], [12,15,8,6]]

del T[3]

for r in T:
    for c in r:
        print(c,end = " ")
    print()
```

Output

When the above code is executed, it produces the following result –

```
11 12 5 2
15 6 10
10 8 12 5
```

 **Print Page**
