Arrays

February 4, 2021

Arrays

[ ]:
```
# Creating Python Arrays
import array as arr
a = arr.array('d', [1.1, 3.5, 4.5])
print(a[0])
```

Accessing Python Array Elements

[ ]:
```
import array as arr
a = arr.array('i', [2, 4, 6, 8])
```

[ ]:
```
print("First element:", a[0])
print("Second element:", a[1])
print("Last element:", a[-1])
```

Changing and Adding Elements

[ ]:
```
import array as arr

numbers = arr.array('i', [1, 2, 3, 5, 7, 10])
```

[ ]:
```
# changing first element
numbers[0] = 0
print(numbers)     # Output: array('i', [0, 2, 3, 5, 7, 10])
```

[ ]:
```
# changing 3rd to 5th element
numbers[2:5] = arr.array('i', [4, 6, 8])
print(numbers)     # Output: array('i', [0, 2, 4, 6, 8, 10])
```

We can add one item to the array using the append() method, or add several items using the extend() method.

[ ]:
```
import array as arr

numbers = arr.array('i', [1, 2, 3])
```

[ ]:
```
numbers.append(4)
print(numbers)     # Output: array('i', [1, 2, 3, 4])
```

[ ]:
```
# extend() appends iterable to the end of the array
numbers.extend([5, 6, 7])
print(numbers)     # Output: array('i', [1, 2, 3, 4, 5, 6, 7])
```

concatenate two arrays using + operator.

[ ]:
```
import array as arr

odd = arr.array('i', [1, 3, 5])
even = arr.array('i', [2, 4, 6])
```

[ ]:
```
numbers = arr.array('i')   # creating empty array of integer
numbers = odd + even
```

[ ]:
```
print(numbers)
```

Removing Python Array Elements We can delete one or more items from an array using Python's del statement.

[ ]:
```
import array as arr

number = arr.array('i', [1, 2, 3, 3, 4])
```

[ ]:
```
del number[2]  # removing third element
print(number)  # Output: array('i', [1, 2, 3, 4])
```

[ ]:
```
del number  # deleting entire array
#print(number)  # Error: array is not defined
```

We can use the remove() method to remove the given item, and pop() method to remove an item at the given index.

[ ]:
```
import array as arr

numbers = arr.array('i', [10, 11, 12, 12, 13])
```

[ ]:
```
numbers.remove(12)
print(numbers)   # Output: array('i', [10, 11, 12, 13])
```

[ ]:
```
print(numbers.pop(2))   # Output: 12
print(numbers)   # Output: array('i', [10, 11, 13])
```

Python Lists Vs Arrays In Python, we can treat lists as arrays. However, we cannot constrain the type of elements stored in a list.

[ ]:
```
import array as arr
# Error
a = arr.array('d', [1, 3.5, "Hello"])
```

Lists are much more flexible than arrays. They can store elements of different data types including strings

The array.array type is just a thin wrapper on C arrays which provides space-efficient storage of basic C-style data types. If you need to allocate an array that you know will not change, then arrays can be faster and use less memory than lists