CDS2003: Data Structures and Object-Oriented Programming

Lecture: Array and List

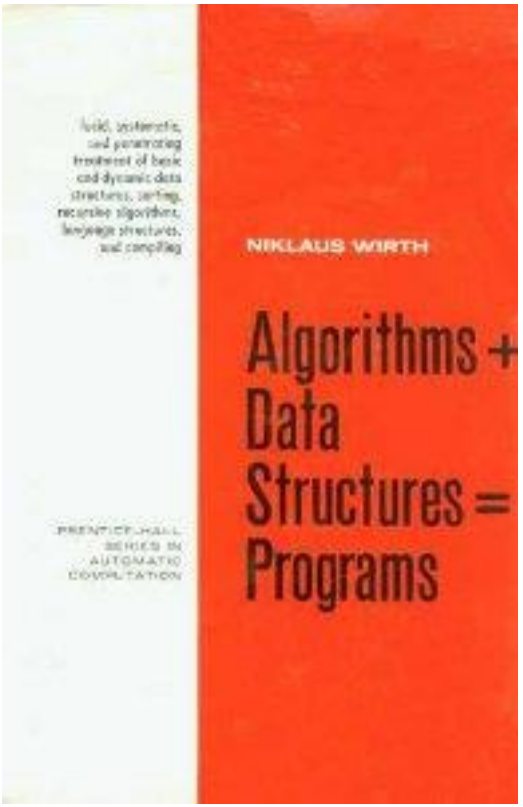
Review

- Complexity theory
  - Complexity of a problem versus complexity of an algorithm/program
  - Class P: P refers to “polynomial time”
  - Class NP: Verifying the correctness of a YES solution in polynomial time
  - P versus NP
  - Class Co-NP
  - NP-hard
  - NP-complete

We shall discuss…

Algorithms + *Data Structures* = *Programs*

-- Niklaus Wirth



- What is data structure?
- What is list/array?

[1] Wirth, N. (1976). Algorithms + Data Structures=programs (Vol. 158). Englewood Cliffs, NJ: Prentice-Hall.

Data structures

- Data structures
  - Any data representation and its associated operation (In the most general sense)
  - Defining how data is organized, stored, and manipulated within a program
  - Fundamental building blocks of programming
  - Organizational tools for data scientists to realize efficient data-driven applications

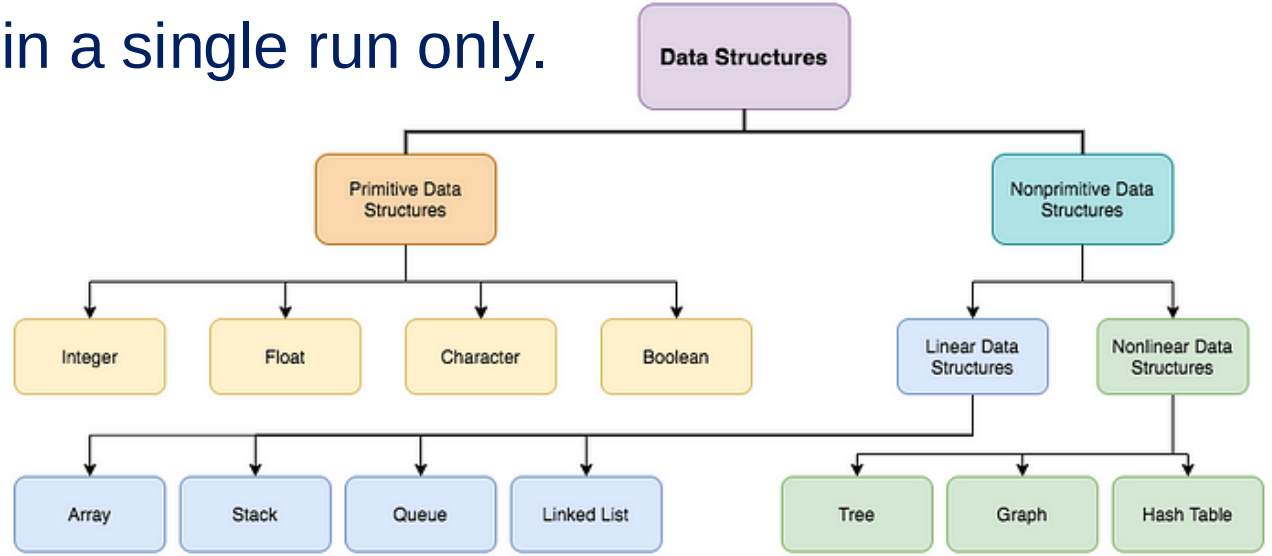
- Primitive data structures
  - Integer; Float; Character; Boolean; Void
- Nonprimitive data structures
  - Array; Linked List; Stack; Queue; Tree; Graph; Hash table

- Realizing the costs and benefits of each data structure
- Selecting the proper data structure for data and algorithms

[1] https://www.geeksforgeeks.org/what-is-data-structure-types-classifications-and-applications/

Nonprimitive Data structures

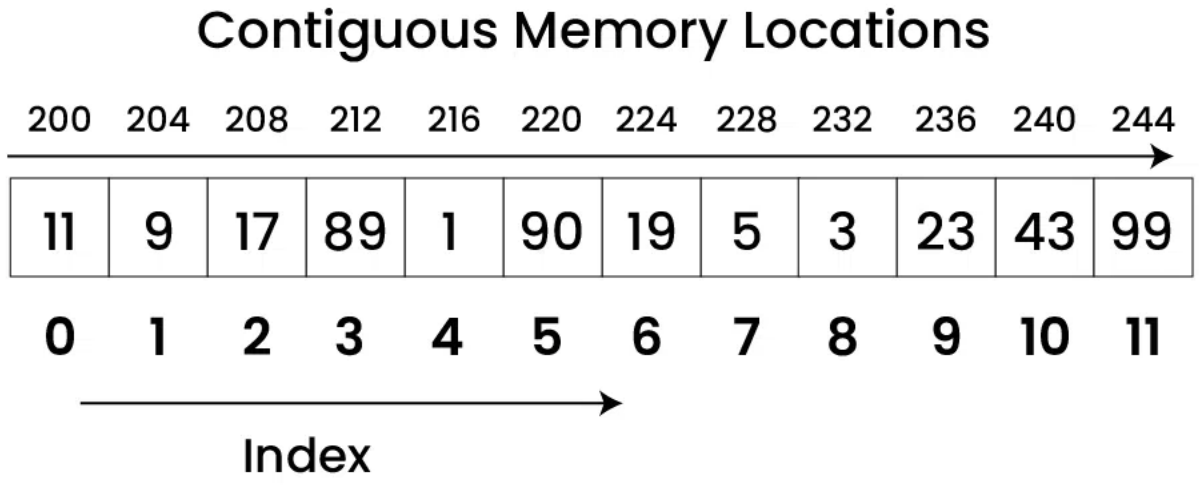
- Linear data structure
  - Data elements are arranged sequentially or linearly and each element is attached to its previous and next adjacent elements
  - Array; stack; queue; linked list
- Non-linear data structure
  - We cannot traverse all the elements in a single run only.
  - Tree; graph; Hash table
- Static data structure
  - Having a fixed memory size
- Dynamic data structure
  - Updating the memory size in the runtime



[1] Data structures: The Basics https://medium.com/swlh/data-structures-the-basics-dc356cb97111

What is array

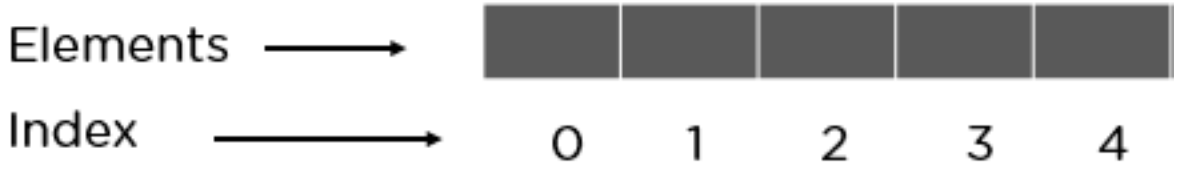
- A linear data structure that collects elements of the same data type and stores them in contiguous and adjacent memory locations.
- Each array element is identified by at least one array index or key.
  - Address
  - Element
  - Index



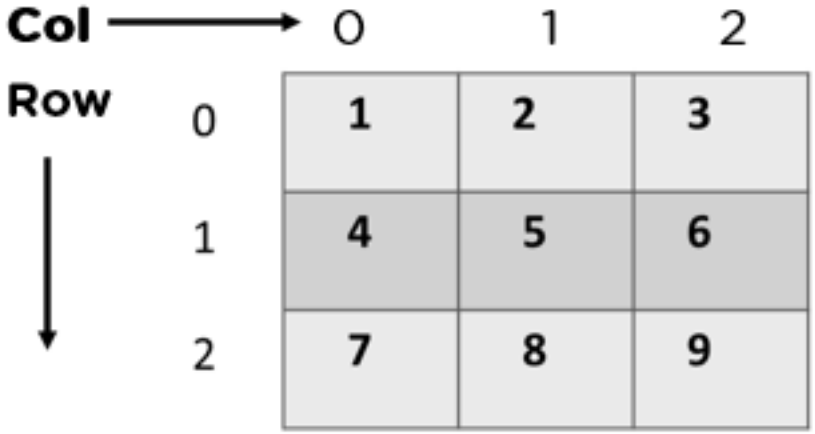
[1] https://www.geeksforgeeks.org/introduction-to-arrays-data-structure-and-algorithm-tutorials/
[2] https://www.simplilearn.com/tutorials/data-structure-tutorial/arrays-in-data-structure

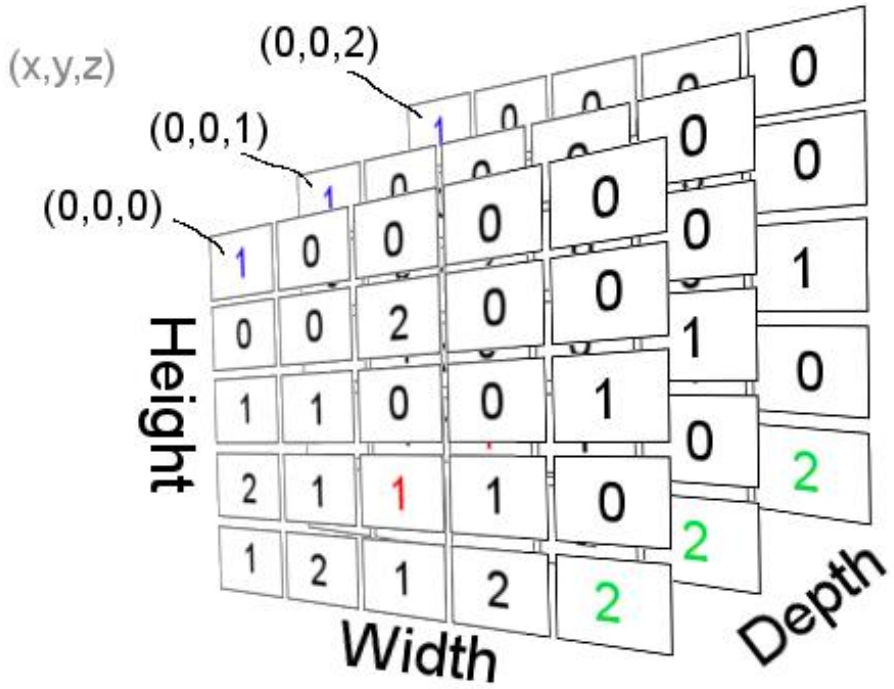
Types of array

- One-dimensional arrays
  - Vectors or lists in Python



- Multi-dimensional array
  - A two-dimensional array (matrix)
  - A three-dimensional array (tensor)





[1] https://www.geeksforgeeks.org/introduction-to-arrays-data-structure-and-algorithm-tutorials/
[2] https://www.simplilearn.com/tutorials/data-structure-tutorial/arrays-in-data-structure
[3] https://www.construct.net/en/tutorials/arrays-beginners-170/3-dimensional-arrays-5

Array in Python

- Array in Python can be created by
  - Importing the array module and using the array(data_type, value_list) function
- Creating an array of three basic types, namely integer, char, and float
  - Time complexity: $O(1)$
  - Space complexity: $O(n)$

```
import array as arr

# creating an array with integer type
a = arr.array('i', [1, 2, 3])
print (type(a), a)

# creating an array with char type
a = arr.array('u', 'BAT')
print (type(a), a)

# creating an array with float type
a = arr.array('d', [1.1, 2.2, 3.3])
print (type(a), a)
```

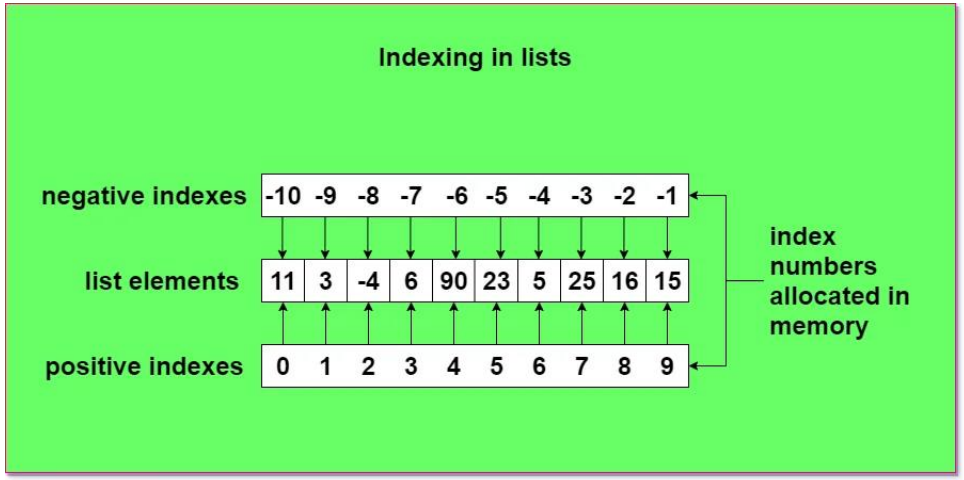
[1] https://www.tutorialspoint.com/python/python_arrays.htm

Basic operations of array

- Accessing elements
- Adding/inserting elements
- Concatenation
- Removing/deleting elements
- Slicing
- Searching elements
- Updating elements
- Counting elements
- Finding the length of the array
- Looping through arrays/array traversal

Basic operations of array

- Accessing elements
  - Time complexity: $O(1)$
  - Space complexity: $O(1)$
- Positive and negative indexing




```
import array as arr
a = arr.array('i', [1, 2, 3, 4, 5, 6])
print("Access element is: ", a[0])
print("Access element is: ", a[3])
b = arr.array('d', [2.5, 3.2, 3.3])
print("Access element is: ", b[1])
print("Access element is: ", b[2])
```

Output:

```
Access element is:  1
Access element is:  4
Access element is:  3.2
Access element is:  3.3
```

[1] https://www.geeksforgeeks.org/python-arrays/
[2] https://www.askpython.com/python/list/negative-indexing

Basic operations of array

- Adding/inserting elements

```
import array as arr
a = arr.array('i', [1, 2, 3])
print("Array before insertion : ", end=" ")
for i in range(0, 3):
    print(a[i], end=" ")
print()

a.insert(1, 4)
print("Array after insertion : ", end=" ")
for i in (a):
    print(i, end=" ")
print()

b = arr.array('d', [2.5, 3.2, 3.3])
print("Array before insertion : ", end=" ")
for i in range(0, 3):
    print(b[i], end=" ")
print()

b.append(4.4)
print("Array after insertion : ", end=" ")
for i in (b):
    print(i, end=" ")
print()
```

Output:

```
Array before insertion :  1 2 3
Array after insertion :  1 4 2 3
Array before insertion :  2.5 3.2 3.3
Array after insertion :  2.5 3.2 3.3 4.4
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Extending elements

```
import array as arr
a=arr.array('i',[1,2,3,4,5,6])
print("The Before extend array is :",end=" ")
for i in range(0,6):
    print(a[i],end=" ")
a.extend([7,8,9,10,11,12])
print("\nThe After extend array is :",end=" ")
for i in range(0,12):
    print(a[i],end=" ")
```

Output:

```
The Before extend array is : 1 2 3 4 5 6
The After extend array is : 1 2 3 4 5 6 7 8 9 10 11 12
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Concatenation

```
import array as arr
a=arr.array('i',[1,2,3,4,5,6])
print("The Before extend array is :",end=" ")
for i in range(0,6):
    print(a[i],end=" ")

b=arr.array('i',[7,8,9,10,11,12])
a = a + b
print("\nThe After extend array is :",end=" ")
for i in range(0,12):
    print(a[i],end=" ")
```

Output:

```
The Before extend array is : 1 2 3 4 5 6
The After extend array is : 1 2 3 4 5 6 7 8 9 10 11 12
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Removing/deleting one or more elements

```
import array
arr = array.array('i', [1, 2, 3, 1, 5])
print("The new created array is : ", end="")
for i in range(0, 5):
    print(arr[i], end=" ")

print("\r")
print("The popped element is : ", end="")
print(arr.pop(2))
print("The array after popping is : ", end="")
for i in range(0, 4):
    print(arr[i], end=" ")

print("\r")
arr.remove(2)
print("The array after removing is : ", end="")
for i in range(0, 3):
    print(arr[i], end=" ")
```

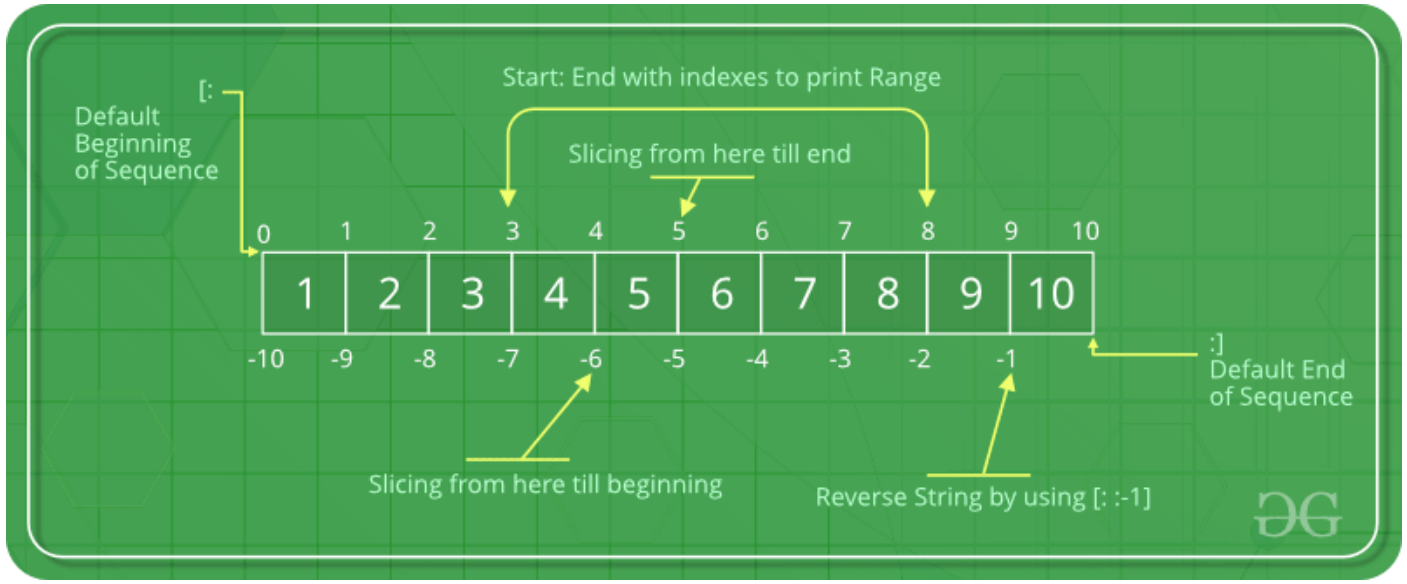
Output:

```
The new created array is : 1 2 3 1 5
The popped element is : 3
The array after popping is : 1 2 1 5
The array after removing is : 1 1 5
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Slicing
  - Printing elements from beginning to a range by [:Index]
  - Printing elements from end by [:-Index]
  - Printing elements from specific Index till the end by [Index:]
  - Printing elements within a range by [Start Index:End Index]
  - Printing the whole array by [:] and the whole array in reverse order by [::-1]



[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Slicing

```
import array as arr
l = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
a = arr.array('i', l)
print("Initial Array: ")
for i in (a):
    print(i, end=" ")
Sliced_array = a[3:8]
print("\nSlicing elements in a range 3-8: ")
print(Sliced_array)
Sliced_array = a[5:]
print("\nElements sliced from 5th element till the end: ")
print(Sliced_array)
Sliced_array = a[:]
print("\nPrinting all elements using slice operation: ")
print(Sliced_array)
```

Output:

```
Initial Array:
1 2 3 4 5 6 7 8 9 10
Slicing elements in a range 3-8:
array('i', [4, 5, 6, 7, 8])

Elements sliced from 5th element till the end:
array('i', [6, 7, 8, 9, 10])

Printing all elements using slice operation:
array('i', [1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Searching an element

```
import array
arr = array.array('i', [1, 2, 3, 1, 2, 5])
print("The new created array is : ", end="")
for i in range(0, 6):
    print(arr[i], end=" ")

print("\r")
print("The index of 1st occurrence of 2 is : ", end="")
print(arr.index(2))
print("The index of 1st occurrence of 1 is : ", end="")
print(arr.index(1))
```

Output:

```
The new created array is : 1 2 3 1 2 5
The index of 1st occurrence of 2 is : 1
The index of 1st occurrence of 1 is : 0
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Updating elements

```
import array
arr = array.array('i', [1, 2, 3, 1, 2, 5])
print("Array before updation : ", end="")
for i in range(0, 6):
    print(arr[i], end=" ")

print("\r")
arr[2] = 6
print("Array after updation : ", end="")
for i in range(0, 6):
    print(arr[i], end=" ")
print()
arr[4] = 8
print("Array after updation : ", end="")
for i in range(0, 6):
    print(arr[i], end=" ")
```

Output:

```
Array before updation : 1 2 3 1 2 5
Array after updation : 1 2 6 1 2 5
Array after updation : 1 2 6 1 8 5
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Counting elements
- Finding the length of the array

```
import array
my_array = array.array('i', [1, 2, 3, 4, 2, 5, 2])
count = my_array.count(2)
print("Number of occurrences of 2:", count)
length = len(my_array)
print("Length of the array:", length)
```

Output:

```
Number of occurrences of 2: 3
Length of the array: 7
```

[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of array

- Looping through arrays/array traversal

```
import array as arr
array_1 = arr.array('i',[7,3,2,6,9])
for i in array_1:
    print(i + 3, end=' ')
```

Output:

10 6 5 9 12

[1] https://www.geeksforgeeks.org/python-arrays/

Python List

- Python Lists are just like array.
- It is flexible as the items in a list do not need to be of the same type.

```
List = [1, 2, 3, "GFG", 2.3, 'GFG', 1]
print(List)

import array as arr
array_1 = arr.array('i', [3, 1, 2])
print(array_1)
```

Output:

```
[1, 2, 3, 'GFG', 2.3, 'GFG', 1]
array('i', [3, 1, 2])
```

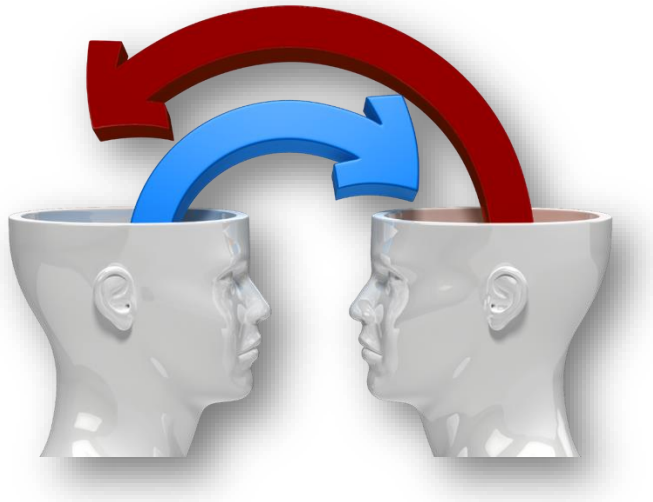
[1] https://www.geeksforgeeks.org/python-arrays/

Basic operations of a Python List

- Getting the size of Python list: len()
- Accessing elements from the List: List[0], List[1], List[-1]
- Taking input of a Python List
- Adding elements to a Python List
- Reversing a List in Python
- Removing elements from the List
- Slicing of a List

[1] https://www.geeksforgeeks.org/python-lists/?ref=header_outind#getting-the-size-of-python-list

Discussion



Q & A!