IT2153/IT2352/IT2553/IT2653/IT2852

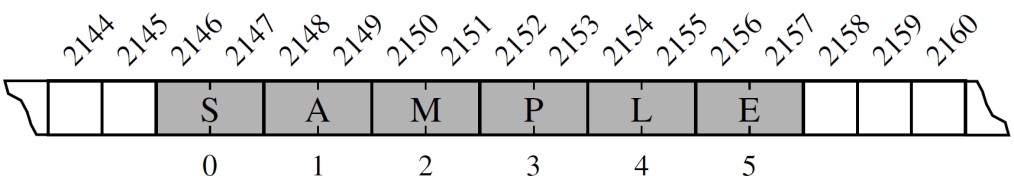
Data Structures & Algorithms



**Tutorial 09**

**Array-Based Sequences**

1. Python’s **compact array** stores a group of related variables (of the same type and size) one after another in a contiguous portion of the computer’s memory.



For example, the above represent an array of six characters that occupies 12 bytes of memory (2 bytes for each character), starting from memory address 2146 to 2157. Each location within the array is a **cell** and we use an integer **index** to describe its location within the array (e.g. the cell of the above array with index 4 has content L and is stored in bytes 2154 and 2155).

Given the memory address at which an array starts, the number of bytes per element, and a desired index within the array. Suggest a formula that can be used to compute the memory address of a specific cell in the array.

***Let memory address at which the array starts 🡪 start***

***Let the number of bytes per element 🡪 cell size***

***Let the desired index within the array 🡪 index***

***Formula: Start + cell size \* index***

1. Refer to Python’s documentation on its array module:

• <https://docs.python.org/3/library/array.html>

Write a Python program that performs the following:

* 1. Create an array of integers, array\_int

import array as arr

array\_int = arr.array()

* 1. Initialise array\_int with the following values [1, 5, 7, 9, 100]

array\_int = arr.array(‘i’, [1,5,7,9,100])

* 1. Insert the number 3 into the **second element** of array\_int

array\_int.insert(1, 3)

* 1. Remove the number 100 from array\_int

array\_int.pop(-1)

* 1. Append the number 11 into array\_int

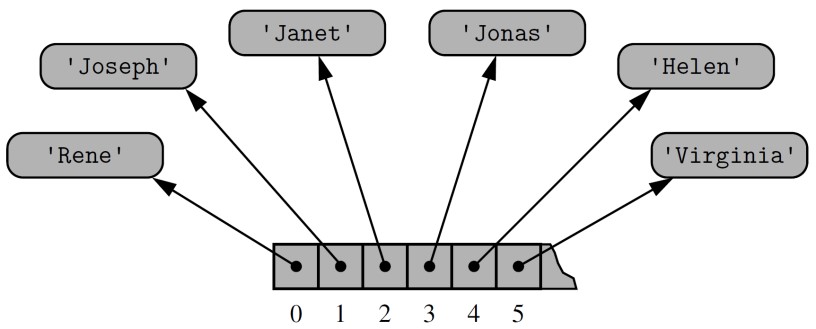
array\_int.insert(11)

* 1. Reverse the order of the numbers in array\_int

array\_int.reverse()

[**NOTE**: Print out the content of array\_int after every step to verify correctness of your code.]

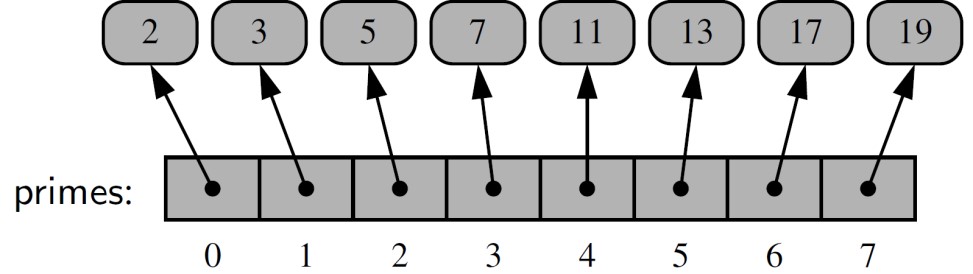
1. The diagram below illustrates how Python uses **referential array** (an array of object references) to represent a list or tuple instance of objects:



At the lowest level, what is stored in the array is a consecutive sequence of memory addresses at which the elements of the sequence reside. Although the relative size of the individual elements may vary, the number of bits to store the memory address of each element is fixed (e.g. 64-bits per memory address).

The diagram below represents a Python list – primes that is declared with the following statement:

primes = [2, 3, 5, 7, 11, 13, 17, 19]



Modify the diagram to reflect the changes (e.g. new lists / objects / references?) after each of the following statements is performed:

* 1. temp = primes[3:6]
* **index: 3, 4, 5**

***Diagram

Description automatically generated***

* 1. temp[2] = 15

Diagram

Description automatically generated

Similarly, with the help of a diagram, describe the changes (e.g. new lists / objects / references?) after each of the following statements is performed:

* 1. counters = [0] \* 8

Diagram

Description automatically generated with medium confidence

* 1. counters[2] += 1

***^^same as counters[2] = counters[2] + 1***

Diagram

Description automatically generated

***-- End of Tutorial --***

AY2020/21 S1 Page 2