Array

Jin Hyun Kim

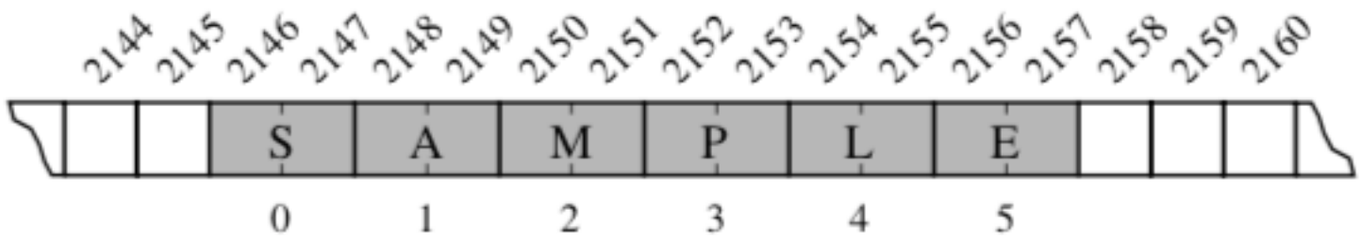
Autumn 2019

Under this topic

- Array
- Referential Array
- Dynamic Array

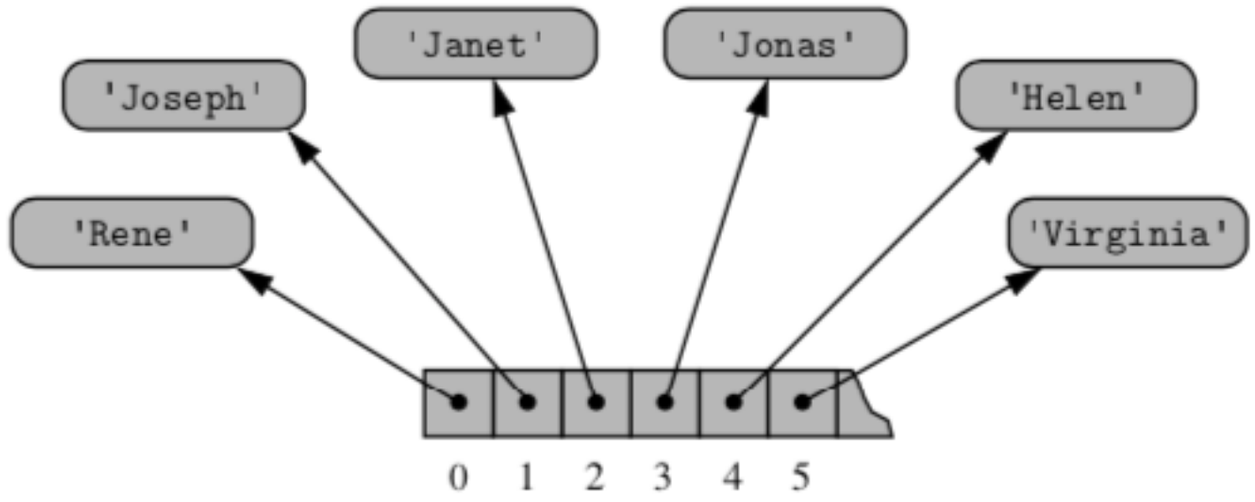
Array

Random Access Memory



A Python string embedded as an array of characters in the computer's memory. We assume that each Unicode character of the string requires two bytes of memory. The numbers below the entries are indices into the string.

Referential Array



Referential Array

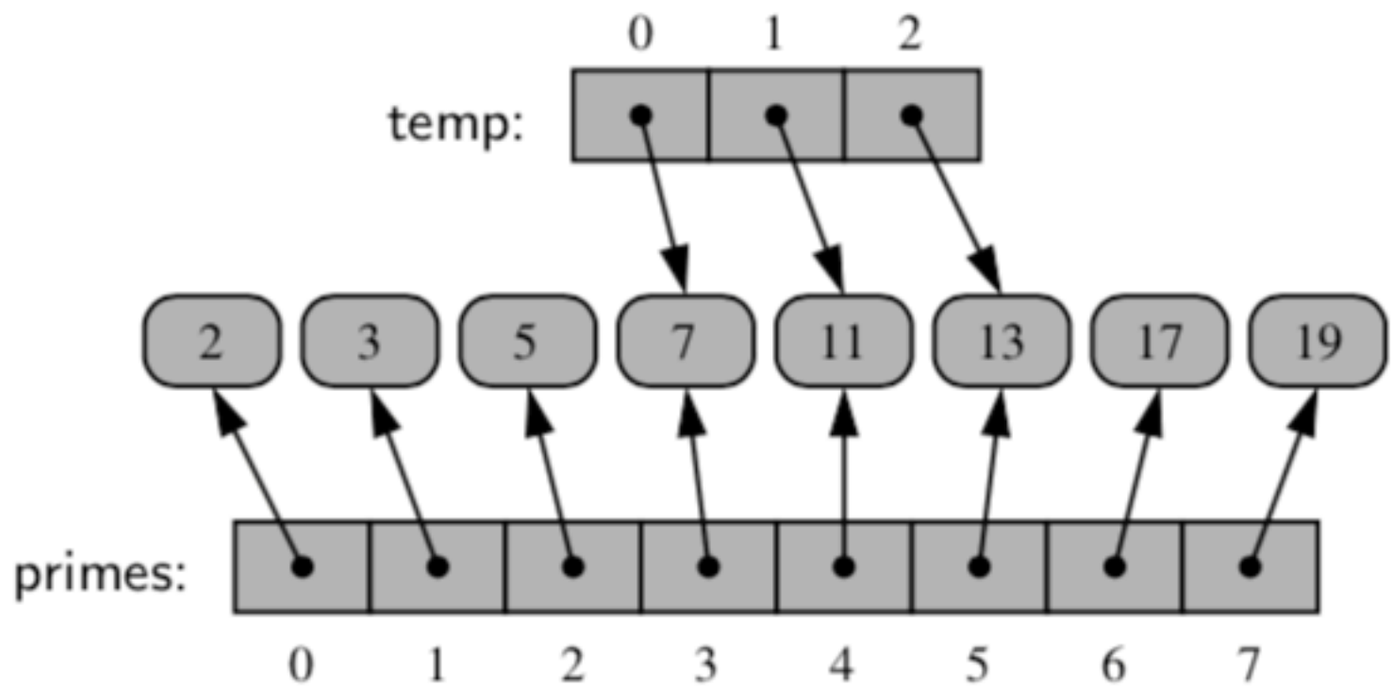


Figure 5.5: The result of the command temp = primes[3:6].

Referential Array

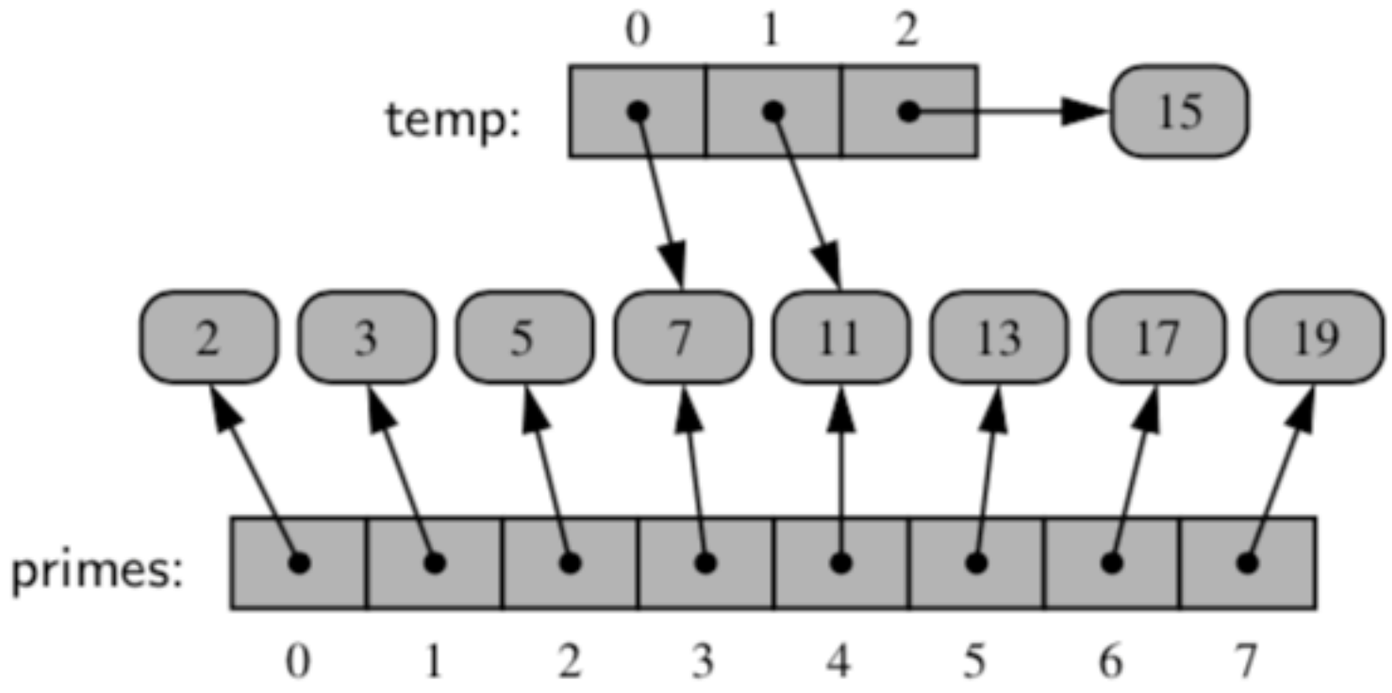


Figure 5.6: The result of the command temp[2] = 15 upon the configuration portrayed in Figure 5.5.

Referential Array

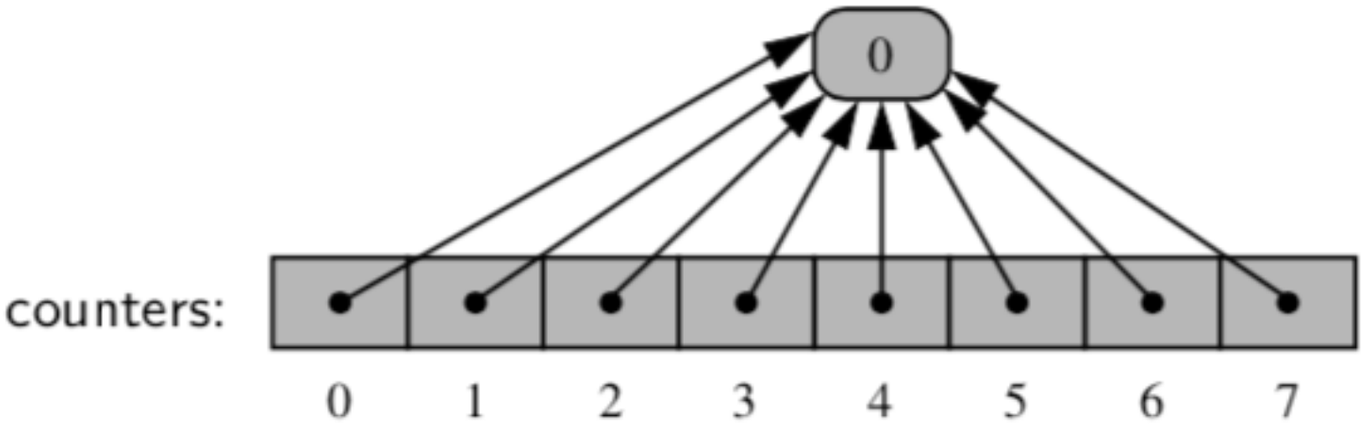


Figure 5.7: The result of the command data $=$ [0] $*$ 8.

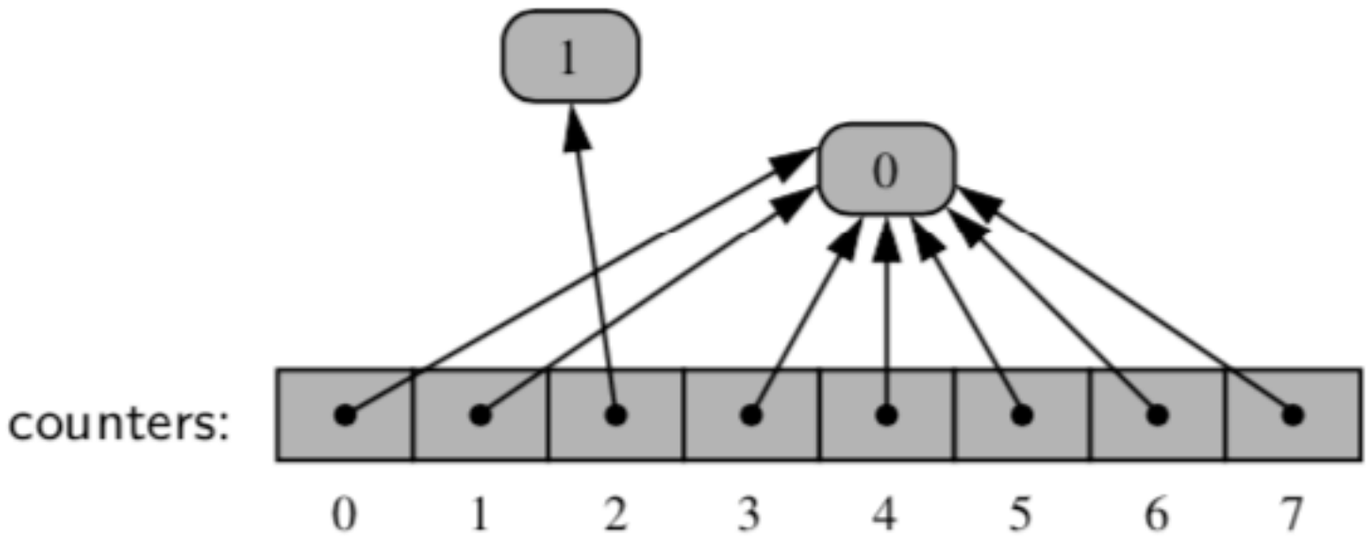


Figure 5.8: The result of command data[2] $+=$ 1 upon the list from Figure 5.7.

Referential Array

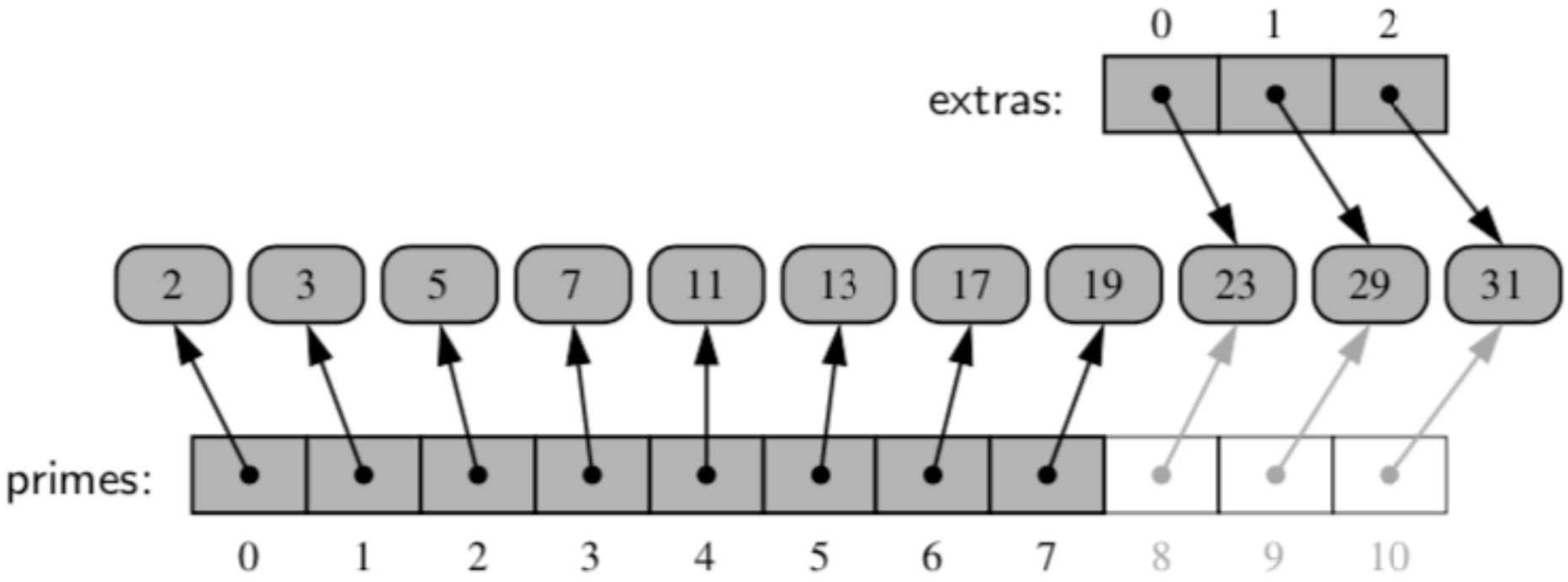


Figure 5.9: The effect of command primes.extend(extras), shown in light gray.

Dynamic Array

```
1  import ctypes                                    # provides low-level arrays
2
3  class DynamicArray:
4    """A dynamic array class akin to a simplified Python list."""
5
6    def __init__(self):
7      """Create an empty array."""
8      self._n = 0                                  # count actual elements
9      self._capacity = 1                           # default array capacity
10     self._A = self._make_array(self._capacity)   # low-level array
11
12   def __len__(self):
13     """Return number of elements stored in the array."""
14     return self._n
15
16   def __getitem__(self, k):
17     """Return element at index k."""
18     if not 0 <= k < self._n:
19       raise IndexError('invalid index')
20     return self._A[k]                            # retrieve from array
21
22   def append(self, obj):
23     """Add object to end of the array."""
24     if self._n == self._capacity:                # not enough room
25       self._resize(2 * self._capacity)           # so double capacity
26     self._A[self._n] = obj
27     self._n += 1
28
29   def _resize(self, c):                          # nonpublic utitity
30     """Resize internal array to capacity c."""
31     B = self._make_array(c)                      # new (bigger) array
32     for k in range(self._n):                     # for each existing value
33       B[k] = self._A[k]
34     self._A = B                                  # use the bigger array
35     self._capacity = c
36
37   def _make_array(self, c):                      # nonpublic utitity
38     """Return new array with capacity c."""
39     return (c * ctypes.py_object)( )             # see ctypes documentation
```

```python
def main():
    da = DynamicArray()
    da.append(1)
    da.append(2)
    da.append(3)
    print(da.__getitem__(2))
    print(da.__getitem_all__())
if __name__ == "__main__":
    main()
```

**** 모든 배열을 리턴하는**
__getitem_all__()을 구현하라.

Next Topic

- Linked List