Datu struktūras

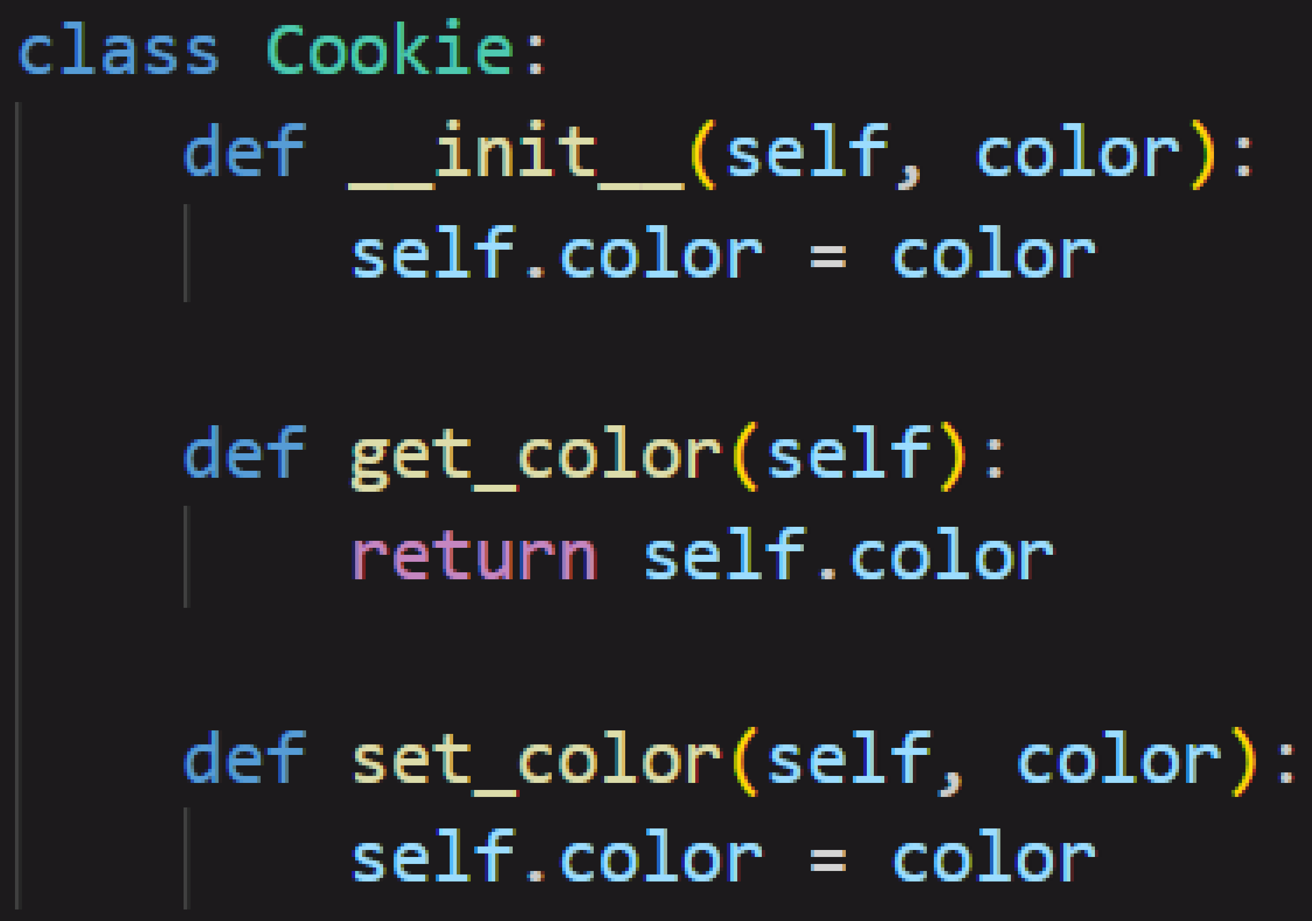
Array

Classes

```python
class Cookie:
    def __init__(self, color):
        self.color = color
```

```python
cookie_one = Cookie('green')
```

```python
cookie_two = Cookie('blue')
```



```python
class Cookie:
    def __init__(self, color):
        self.color = color

    def get_color(self):
        return self.color

    def set_color(self, color):
        self.color = color
```

class

cookie_one

cookie_two

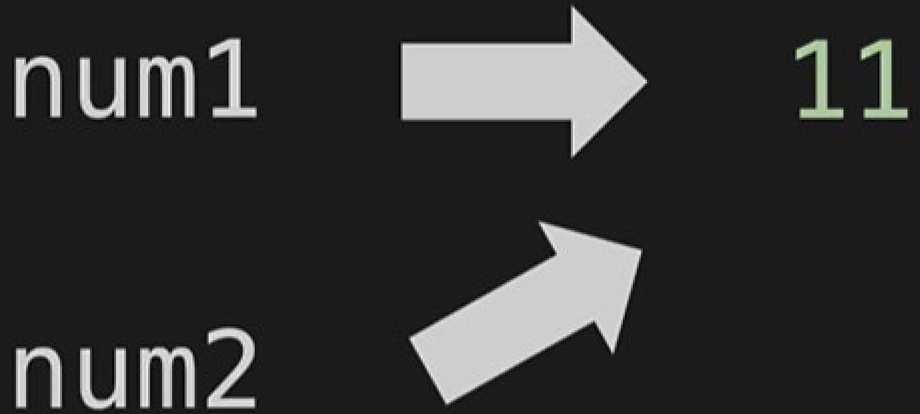
```python
class LinkedList:
    def __init__(self, value):

    def append(self, value):

    def pop(self):

    def prepend(self, value):

    def insert(self, index, value):

    def remove(self, index):
```

Pointers

```
num1 = 11

num2 = num1
```

num1 → 11

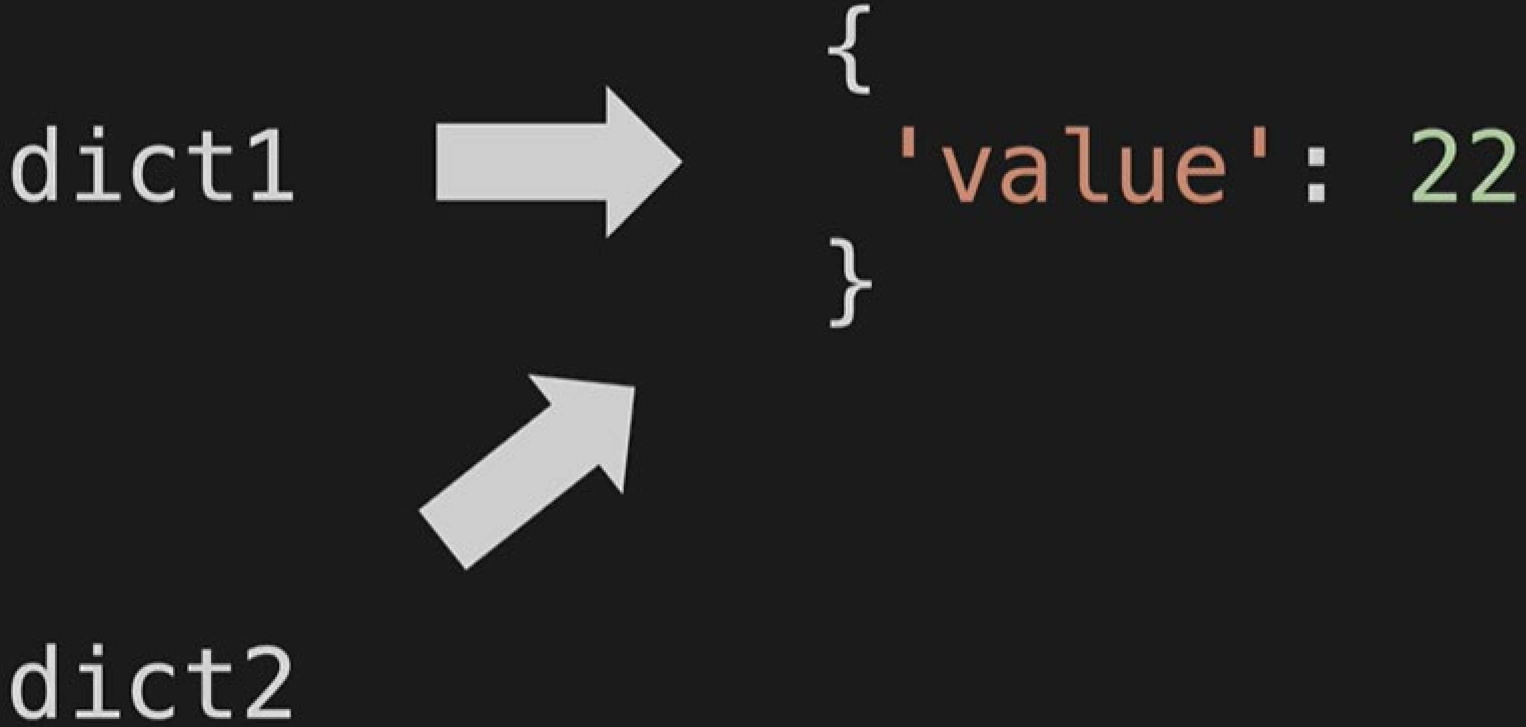
```
num2 = num1
```



num1

11

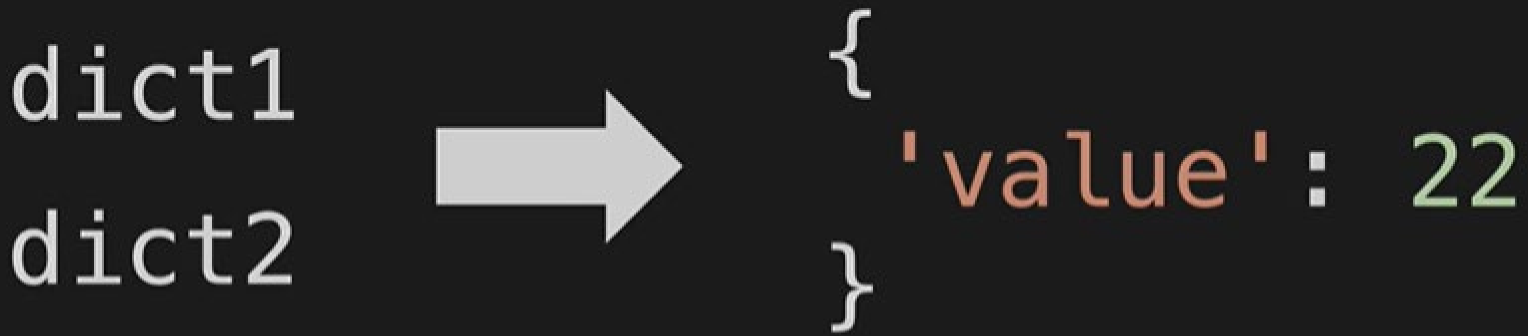
num2



dict1

```
{
  'value': 22
}
```

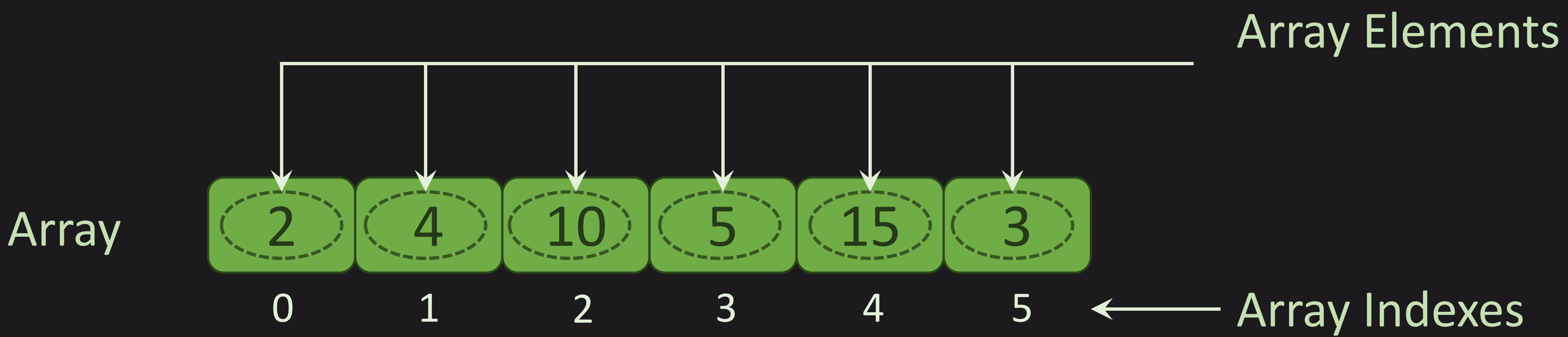
dict2

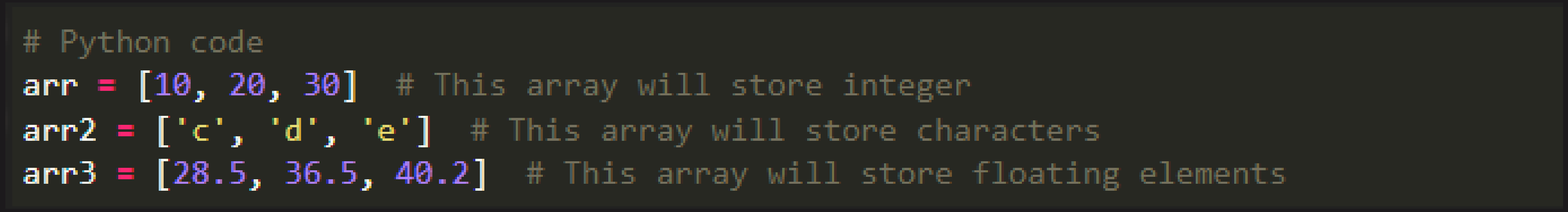


dict1

dict2

```
{
  'value': 22
}
```



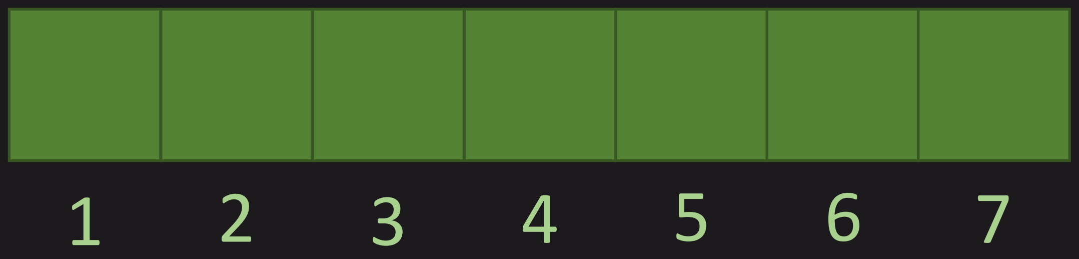


```python
# Python code
arr = [10, 20, 30]  # This array will store integer
arr2 = ['c', 'd', 'e']  # This array will store characters
arr3 = [28.5, 36.5, 40.2]  # This array will store floating elements
```

Definition

Array:

Contiguous area of memory consisting of equal-size elements indexed by contiguous integers.

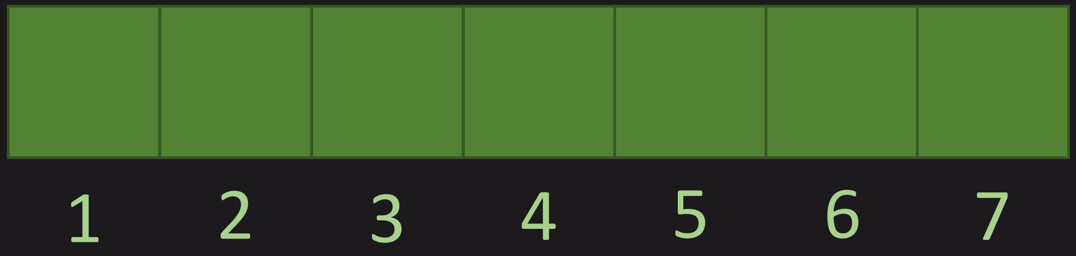


What’s Special About Arrays?

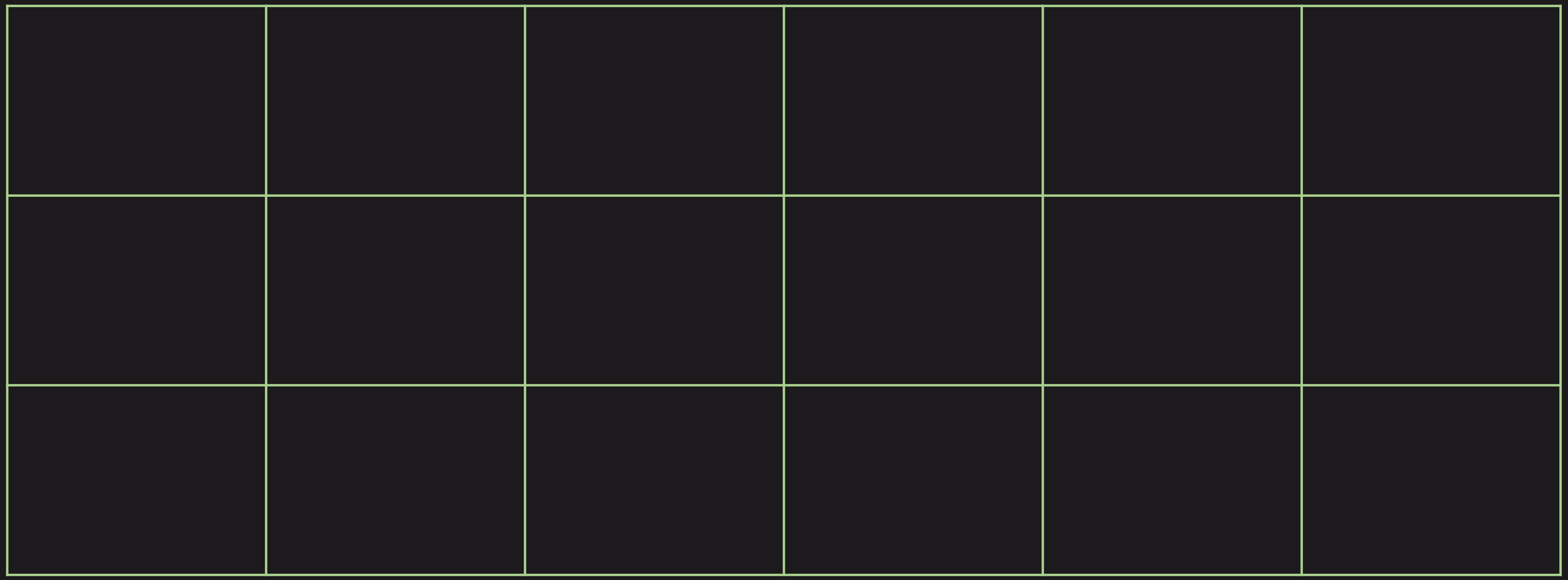
Constant-time access

To read
To write

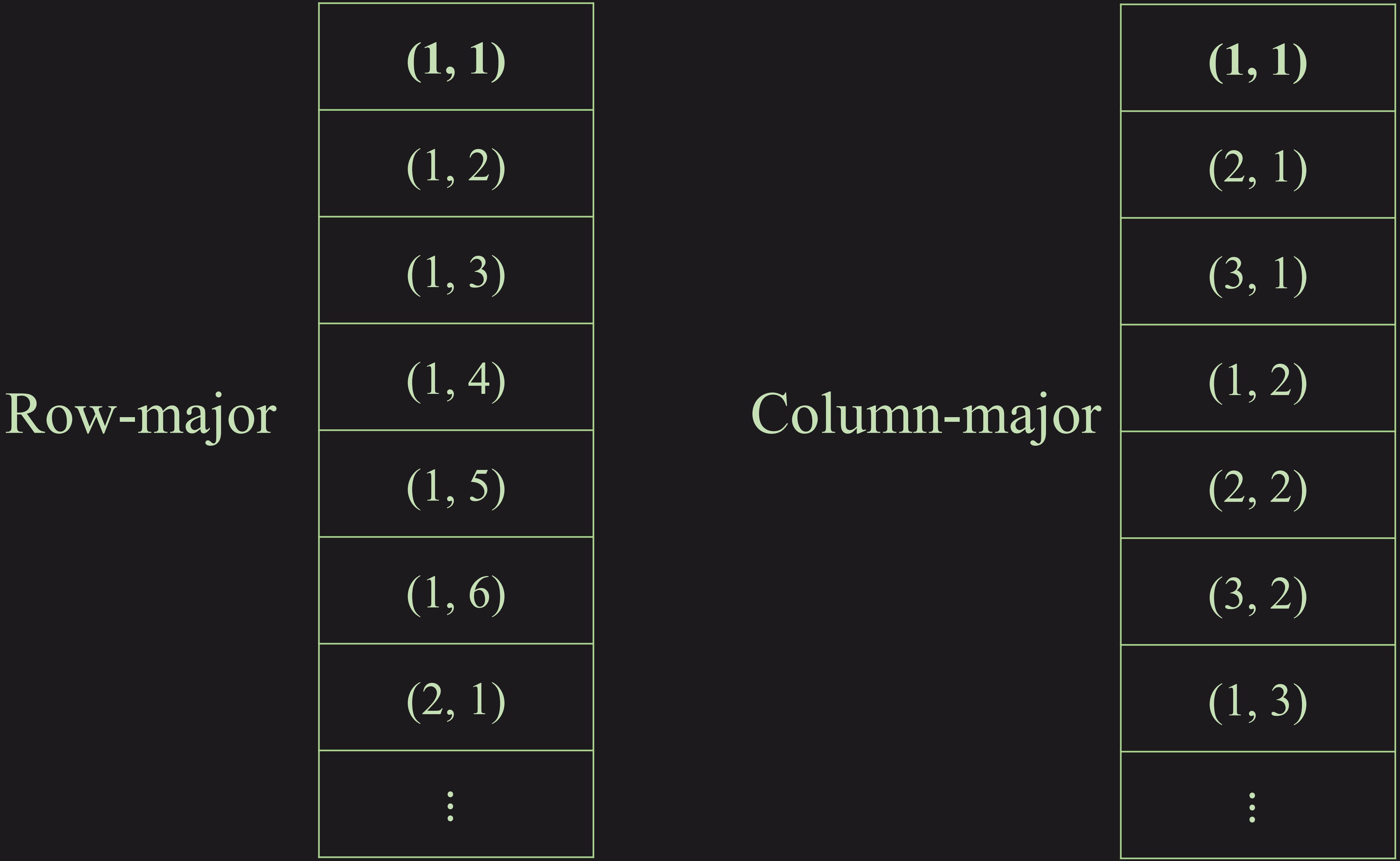
$$\text{array\_addr} + \text{elem\_size} \times (i - \text{first\_index})$$



Multi-Dimensional Arrays



$$\text{array\_addr} + \text{elem\_size} \times ((3 - 1) \times 6 + (4 - 1))$$



Row-major

(1, 1)

(1, 2)

(1, 3)

(1, 4)

(1, 5)

(1, 6)

(2, 1)

⋮

Column-major

(1, 1)

(2, 1)

(3, 1)

(1, 2)

(2, 2)

(3, 2)

(1, 3)

⋮

Times for Common Operations

	Add	Remove
Beginning		
End	$O(1)$	
Middle		

5	8	3	12	4		

Times for Common Operations

	Add	Remove
Beginning		
End	$O(1)$	$O(1)$
Middle		

5	8	3	12			

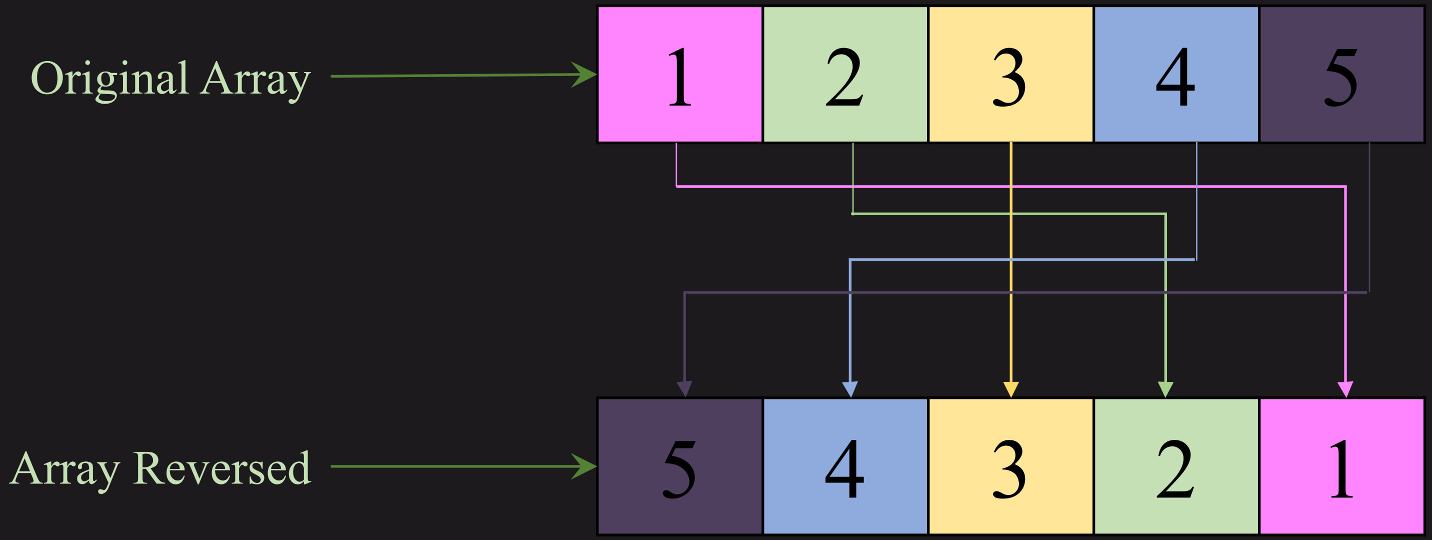
Times for Common Operations

	Add	Remove
Beginning	$O(n)$	$O(n)$
End	$O(1)$	$O(1)$
Middle	$O(n)$	$O(n)$

8	3	12				

Array Reverse

C, C++, Java, Python, Javascript



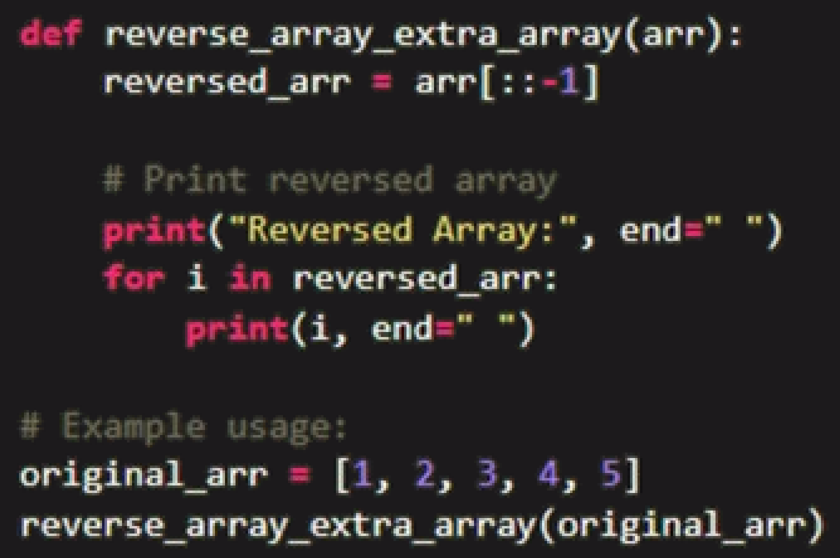
Original Array

1	2	3	4	5

Array Reversed

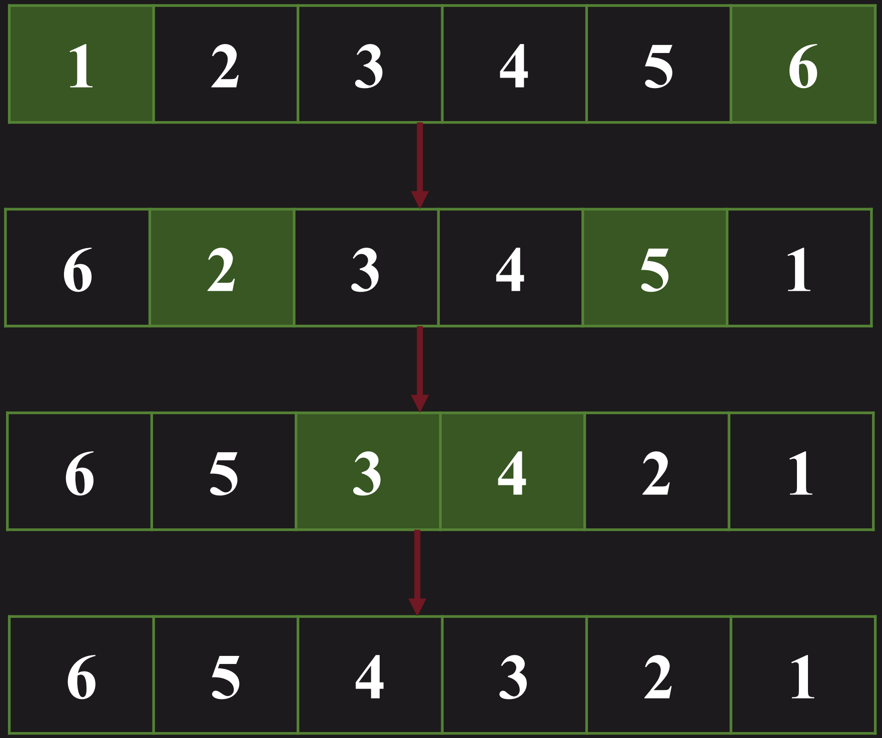
5	4	3	2	1

Array Reverse Using an Extra Array



```python
def reverse_array_extra_array(arr):
    reversed_arr = arr[::-1]

    # Print reversed array
    print("Reversed Array:", end=" ")
    for i in reversed_arr:
        print(i, end=" ")

# Example usage:
original_arr = [1, 2, 3, 4, 5]
reverse_array_extra_array(original_arr)
```

Array Reverse Using a Loop

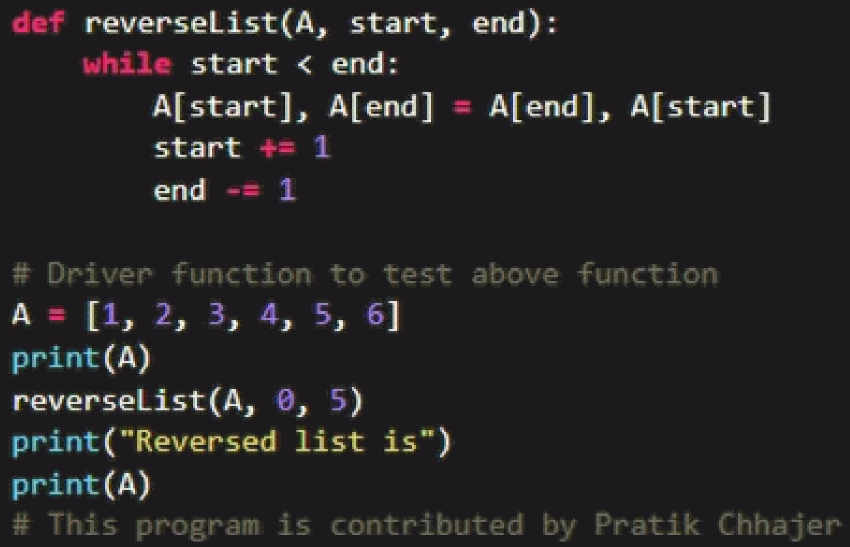


1 2 3 4 5 6

6 2 3 4 5 1

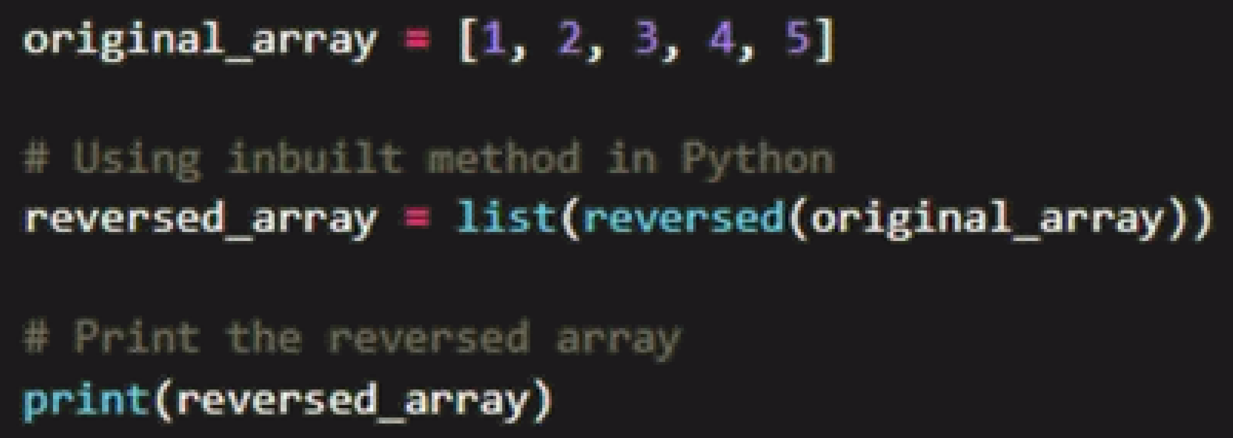
6 5 3 4 2 1

6 5 4 3 2 1



```
def reverseList(A, start, end):
    while start < end:
        A[start], A[end] = A[end], A[start]
        start += 1
        end -= 1

# Driver function to test above function
A = [1, 2, 3, 4, 5, 6]
print(A)
reverseList(A, 0, 5)
print("Reversed list is")
print(A)
# This program is contributed by Pratik Chhajer
```

Array Reverse Inbuilt Methods



```
original_array = [1, 2, 3, 4, 5]

# Using inbuilt method in Python
reversed_array = list(reversed(original_array))

# Print the reversed array
print(reversed_array)
```

Maximum and minimum of an array

Input arr[] = {3, 5, 4, 1, 9}

Output: Minimum element is: 1

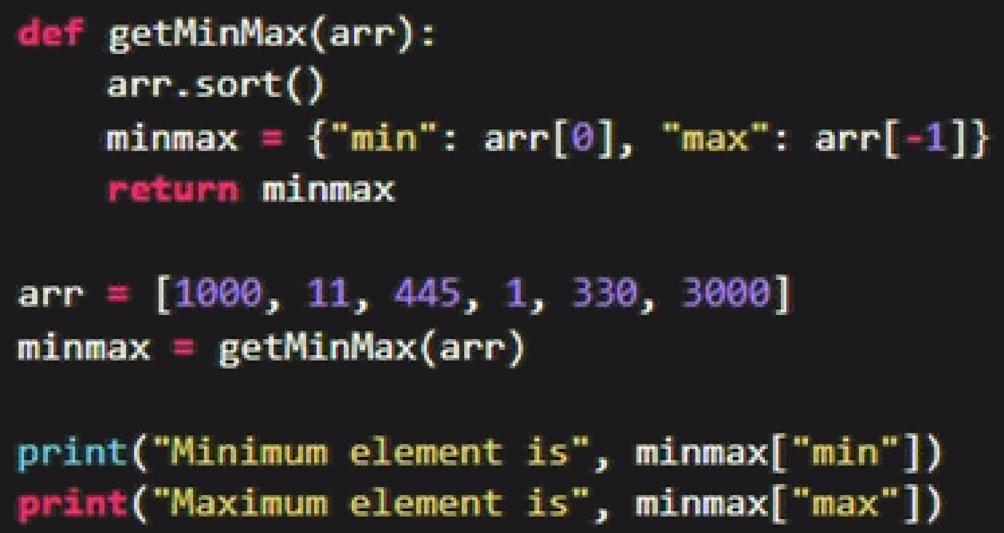
Maximum element is: 9

Input: arr[] = {22, 14, 8, 17, 35, 3}

Output: Minimum element is: 3

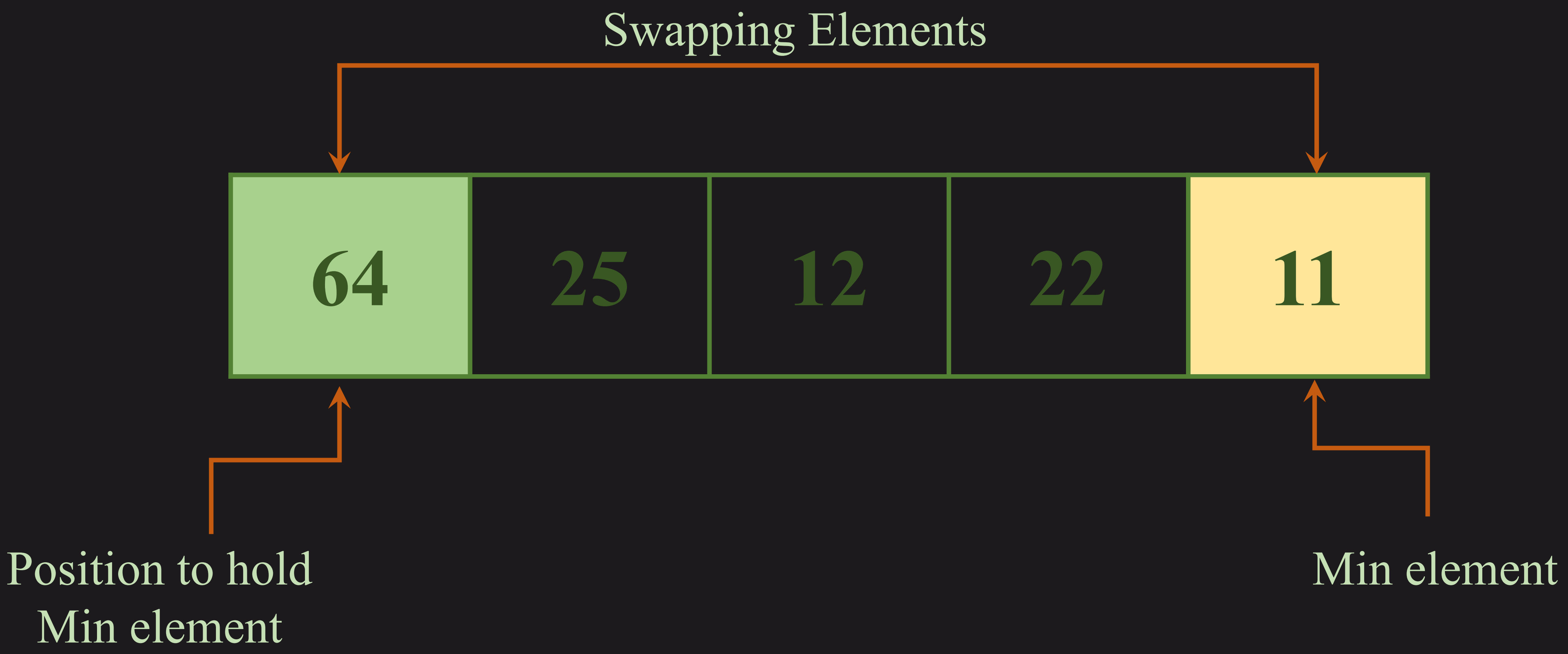
Maximum element is: 35

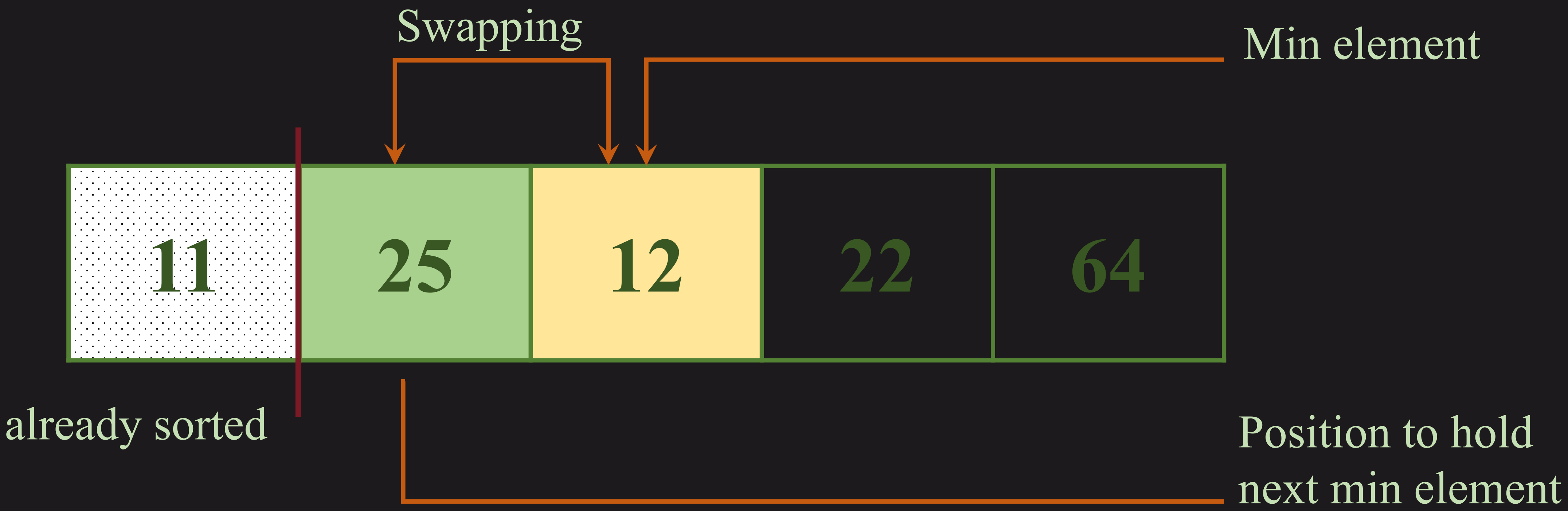
Maximum and minimum of an array using *Sorting*

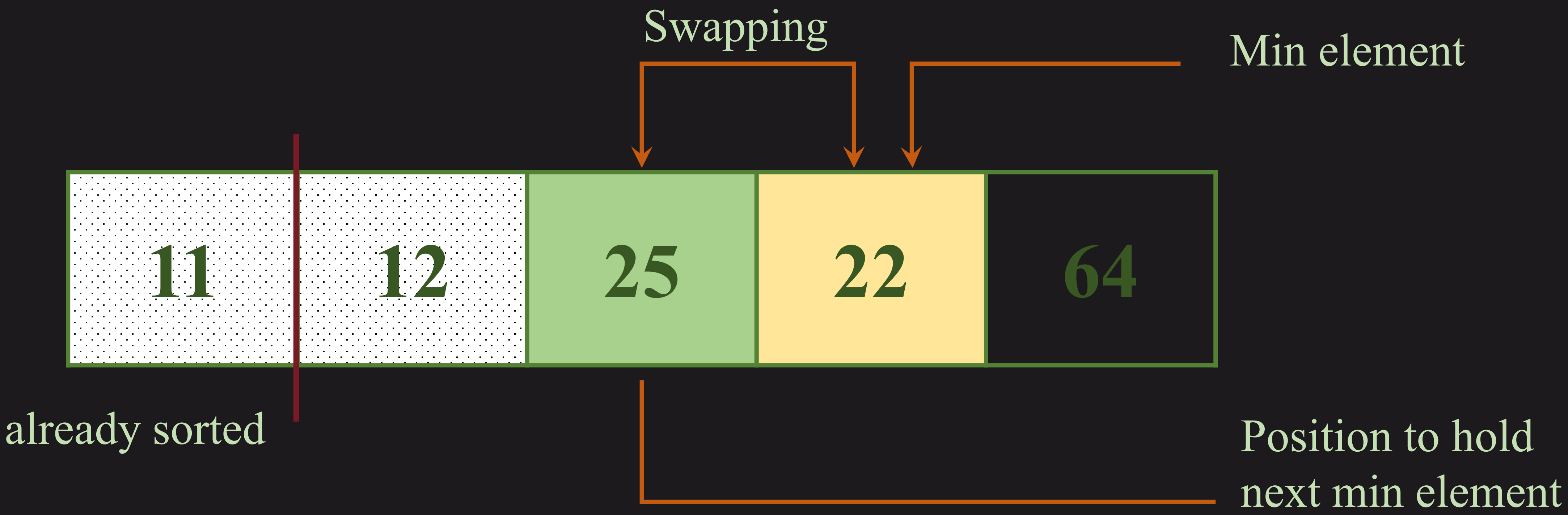


```python
def getMinMax(arr):
    arr.sort()
    minmax = {"min": arr[0], "max": arr[-1]}
    return minmax

arr = [1000, 11, 445, 1, 330, 3000]
minmax = getMinMax(arr)

print("Minimum element is", minmax["min"])
print("Maximum element is", minmax["max"])
```

How does Selection Sort Algorithm work?




Swapping
Min element
11
25
12
22
64
already sorted
Position to hold
next min element



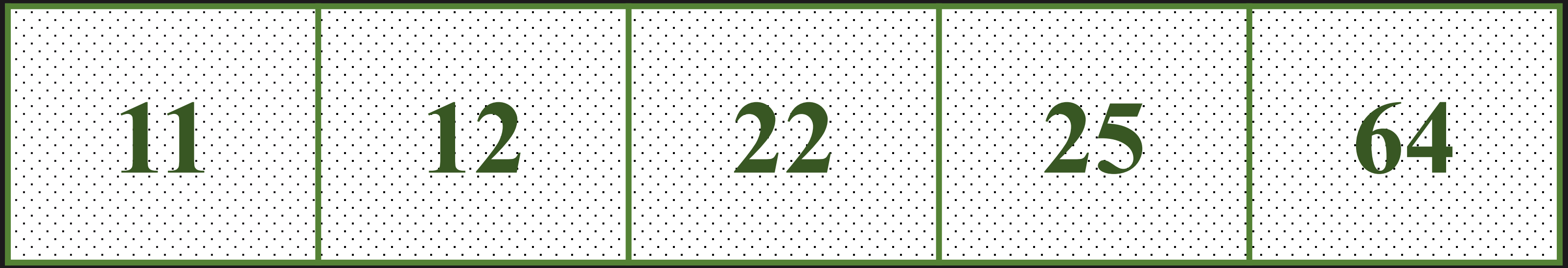
Swapping

Min element

11	12	25	22	64

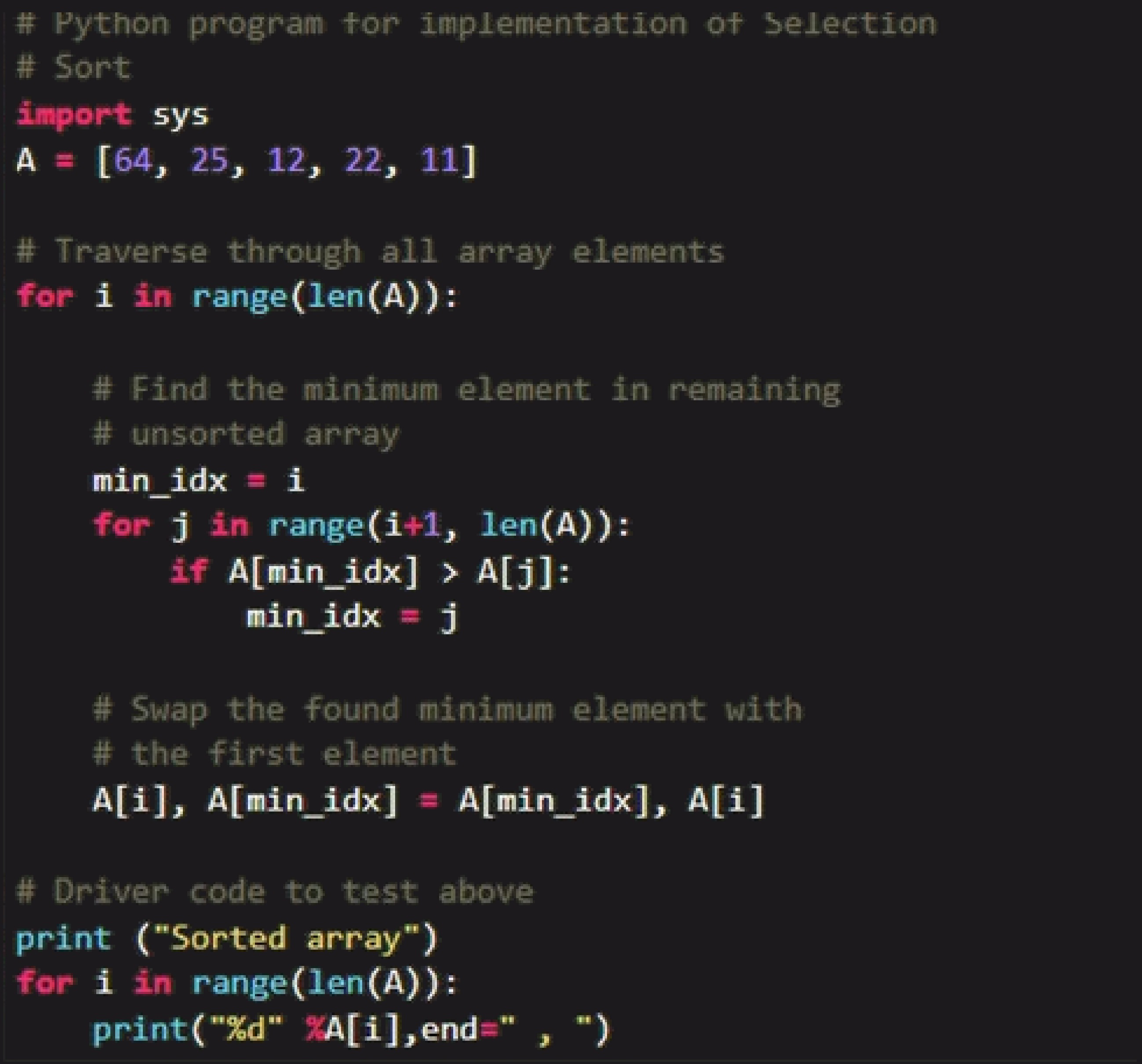
already sorted

Position to hold
next min element



11	12	22	25	64

Sorted array



```
# Python program for implementation of Selection
# Sort
import sys
A = [64, 25, 12, 22, 11]

# Traverse through all array elements
for i in range(len(A)):

    # Find the minimum element in remaining
    # unsorted array
    min_idx = i
    for j in range(i+1, len(A)):
        if A[min_idx] > A[j]:
            min_idx = j

    # Swap the found minimum element with
    # the first element
    A[i], A[min_idx] = A[min_idx], A[i]

# Driver code to test above
print ("Sorted array")
for i in range(len(A)):
    print("%d" %A[i],end=" , ")
```