CSCI-SHU 210 Data Structures

Recitation 4   Array Based Sequences and Dynamic Array

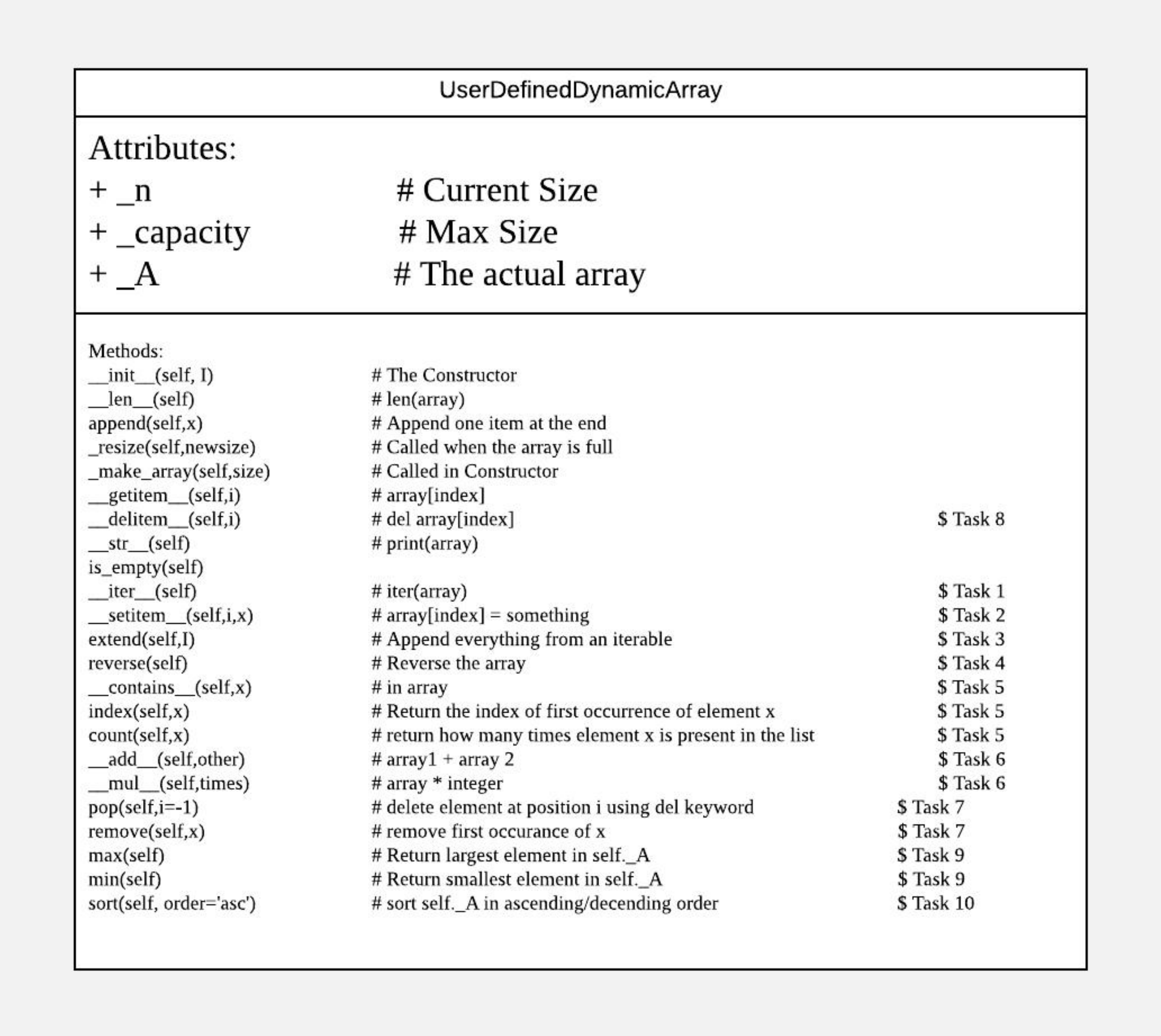
You have a series of tasks in front of you. Complete them! Everyone should code on their own computer, but you are encouraged to talk to others, and seek help from each other and from the Professor/TA/LA.

Important:

- **Understand what is a “low-level array”**
  - Also called “static array”, “compact array”
  - Fixed capacity, continuous chuck of memory, each cell stores the same type.
  - Supports indexing in O(1) time.
- **Understand what is a “dynamic array”**
  - Supports append( ), pop( ) in O(1) amortized time.
  - Capacity can grow and shrink.
- **Understand what is a “python list”**
  - Each cell can store different type. How?

Question 1 (Implement a Dynamic Array)

: /



UserDefinedDynamicArray

Attributes:

+ _n	# Current Size
+ _capacity	# Max Size
+ _A	# The actual array

Methods:

__init__(self, I)	# The Constructor	
__len__(self)	# len(array)	
append(self,x)	# Append one item at the end	
_resize(self,newsize)	# Called when the array is full	
_make_array(self,size)	# Called in Constructor	
__getitem__(self,i)	# array[index]	
__delitem__(self,i)	# del array[index]	$ Task 8
__str__(self)	# print(array)	
is_empty(self)		
__iter__(self)	# iter(array)	$ Task 1
__setitem__(self,i,x)	# array[index] = something	$ Task 2
extend(self,I)	# Append everything from an iterable	$ Task 3
reverse(self)	# Reverse the array	$ Task 4
__contains__(self,x)	# in array	$ Task 5
index(self,x)	# Return the index of first occurrence of element x	$ Task 5
count(self,x)	# return how many times element x is present in the list	$ Task 5
__add__(self,other)	# array1 + array 2	$ Task 6
__mul__(self,times)	# array * integer	$ Task 6
pop(self,i=-1)	# delete element at position i using del keyword	$ Task 7
remove(self,x)	# remove first occurance of x	$ Task 7
max(self)	# Return largest element in self._A	$ Task 9
min(self)	# Return smallest element in self._A	$ Task 9
sort(self, order='asc')	# sort self._A in ascending/decending order	$ Task 10

Figure 1: The `UserDefinedDynamicArray` Class UML Diagram.

: `UserDefinedDynamicArray`
`DynamicArray.py`

:

1: :
`__iter__`

- `__str__` `__str__` `__iter__`
- `__iter__` `yield`
- , `__iter__` `for xxx in list`

```
>>> l = [1,2,3,4]
>>> b = l.__iter__()
>>> b
<list_iterator object at 0x10ade4208>
```

2:

__setitem__

- __delitem__(self, i) del
- __setitem__(self, i, x) list[index] = value
- : __delitem__ __setitem__
    -

```
>>> l = [1,2,3,4]
>>> del l[0:2]
>>> l
[3, 4]
>>> l[1] = 99
>>> l
[3, 99]
```

3:

extend(self,I)

- I self._A

```
>>> l = [1,2,3,4]
>>> l2 = [4,5,6]
>>> l.extend(l2)
>>> l
[1, 2, 3, 4, 4, 5, 6]
```

4:

reverse(self) method

- self._A

```
>>> l = [1,2,3,4]
>>> l.reverse()
>>> l
[4, 3, 2, 1]
```

5:

__contains__(self, x) index(self, x) count(self, x)

- __contains__(self, x) x
  __contains__ in
- index(self, x) x x
  x
- count(self, x) x
  x 0

```
>>> l = [1,2,3,4,1]
>>> 1 in l
True
>>> l.index(1)
0
>>> l.count(1)
2
```

6: Array + Array Array * Integer

__add__(self,other) __mul__(self,times)

- __add__ UserDefinedDyamicArray
  myArray1+myArray2 UserDefinedDyamicArray
  myArray1 myArray2
- __mul__ UserDefinedDyamicArray
  myArray1*3 UserDefinedDyamicArray myArray1

```
>>> l1 = [1,2,3,4]
>>> l2 = [4,5,6]
>>> l1 + l2
[1, 2, 3, 4, 4, 5, 6]
>>> l1 * 3
[1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4]
```

7: pop(i) remove(value)

pop(i) remove(value)

- pop(i)
  del
- i i
- remove()
- __delitem__()
- remove(x) x x
  x

```
>>> l1 = [1,2,3,4,1]
>>> l1.pop(2) # Pop index 2
3
>>> l1.remove(1)  # Remove first occurrence of value 1
>>> print(l1)
[2, 4, 1]
```

8: __delitem__

- __delitem__(self, i)
-

```
>>> l1 = [20,40,60,80,100,120,140,160,180,200]
>>> print(l1, “capacity:”, l1._capacity)
[20,40,60,80,100,120,140,160,180,200] capacity: 16
>>> for i in range(7):
>>>         del l1[0]
>>> print(l1, “capacity:”, l1._capacity)
[160,180,200] capacity: 8
```

9: Max/Min

- max(self) self._A
- min(self) self._A.

```
>>> l1 = [4,7,3,1,9]
>>> l1.max()
9
>>> l1.min()
1
```

10: `UserDefinedDynamicArray` `sort(self, order = ‘asc’)`

-
-

```
>>> l1 = [4,7,3,1,9]
>>> l1.sort()
>>> l1
[1, 3, 4, 7, 9]
>>> l1.sort(order = ‘desc’)
>>> l1
[9, 7, 4, 3, 1]
```