

## CMPSC 462 – Lab-2 (25 points)

### Arrays

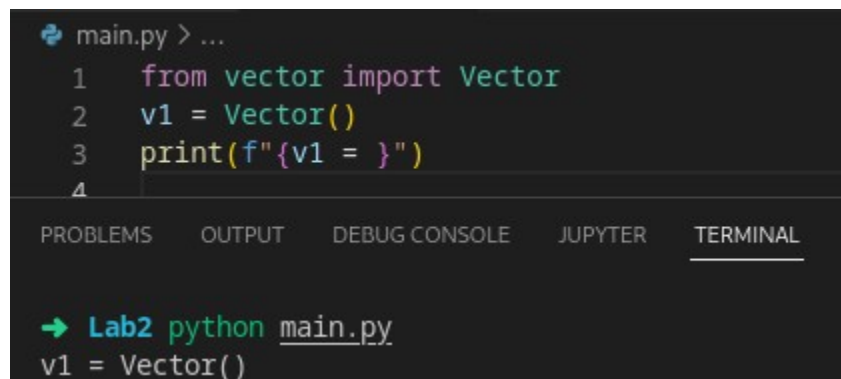
Due date: 9/13/2022

**Note:** attach screenshots of your program and results under each programming exercises. Please make sure that the screenshot is readable. Don't attach a very small screenshot image.

#### Lab Exercise:

While Python provides the built-in list type for constructing and managing mutable sequences, many languages do not provide such a structure, at least not as part of the language itself. To help in further understanding how Python's built-in list works, implement the Vector ADT using the Array class implemented in the chapter. Your implementation should produce a mutable sequence type that works like Python's list structure. When the underlying array needs to be expanded, the new array should double the size of the original. The operations that can be performed on the ADT are described below.


- `Vector()`: Creates a new empty vector with an initial capacity of two elements.



```
main.py > ...
1  from vector import Vector
2  v1 = Vector()
3  print(f"{v1 = }")
4
PROBLEMS  OUTPUT  DEBUG CONSOLE  JUPYTER  TERMINAL

→ Lab2 python main.py
v1 = Vector()
```

- `length()`: Returns the number of items contained in the vector.



```
main.py > ...
4
5  print(f"{v1.length() = }")
6
PROBLEMS  OUTPUT  DEBUG CONSOLE  JUPYTER  TERMINAL

→ Lab2 python main.py
v1.length() = 0
```

- contains ( item ): Determines if the given item is contained in the vector.

```
0
7  print(f"{v1.contains(0) = }")
8
```

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

→ Lab2 python main.py  
v1.contains(0) = False

- getitem ( ndx ): Returns the item stored in the ndx element of the list. The value of ndx must be within the valid range.

```
main.py > ...
8
9  print(f"{v1.getitem(0) = }")
10
```

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

→ Lab2 python main.py  
v1.getitem(0) = None

- setitem ( ndx, item ): Sets the element at position ndx to contain the given item. The value of ndx must be within the valid range, which includes the first position past the last item.

```
main.py > ...
10
11  v1.setitem(index = 0, item = 2)
12  print(f"setitem: {v1 = }")
13
```

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

→ Lab2 python main.py  
setitem: v1 = Vector(2)

- append( item ): Adds the given item to the end of the list.

```
main.py > ...
13
14 v1.append(10)
15 print(f"append: {v1 = }")
16

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

→ Lab2 python main.py
append: v1 = Vector(2, 10)
```

- insert(nd

x, item): Inserts the given item in the element at position ndx. The items in the elements at and following the given position are shifted down to make room for the new item. ndx must be within the valid range.

```
main.py > ...
17 v1.insert(index = 1, item = 1)
18 print(f"insert: {v1 = }")
19

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

→ Lab2 python main.py
insert: v1 = Vector(2, 1, 10)
```

- remove( ndx ): Removes and returns the item from the element from the given ndx position. The items in the elements at and following the given position are shifted up to close the gap created by the removed item. ndx must be within the valid range.

```
main.py > ...
19
20 v1.remove(0)
21 print(f"remove: {v1 = }")
22

PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL

→ Lab2 python main.py
remove: v1 = Vector(1, 10)
```

- indexOf( item ): Returns the index of the vector element containing the given item. The item must be in the list.

```
main.py > ...  
23 print(f"{v1.indexOf(10) = }")  
24  
PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL  
→ Lab2 python main.py  
v1.indexOf(10) = 1
```

- `extend( otherVector )`: Extends this vector by appending the entire contents of the `otherVector` to this vector.

```
main.py > ...  
24  
25 v2 = Vector([1, 2, 3, 4, 5, 6, 7, 8])  
26 v1.extend(v2)  
27 print(f"extend: {v1 = }")  
28  
PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL  
→ Lab2 python main.py  
extend: v1 = Vector(1, 10, 1, 2, 3, 4, 5, 6, 7, 8)
```

- `subVector( from, to )`: Creates and returns a new vector that contains a subsequence of the items in the vector between and including those indicated by the given `from` and `to` positions. Both the `from` and `to` positions must be within the valid range.

```
29 sub = v1.subVector(1, 6)  
30 print(f"subVector: {sub = }")  
  
PROBLEMS OUTPUT DEBUG CONSOLE JUPYTER TERMINAL  
→ Lab2 python main.py  
subVector: sub = Vector(10, 1, 2, 3, 4, 5)
```