Algorithms and Data Structures for Data Science

PyReview and Lists

CS 277
Brad Solomon

January 25, 2021

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN
Department of Computer Science

Lab_fundamentals Feedback

Over 100% of the class (by numbers) got an 100%

Survey results:

- Took ~2-3 hours on average to complete
- Class lecture not helpful for completing assignment
- No consensus on whether lab improved programming confidence
- Request to slow class down

Learning Objectives

Practice coding through in class review exercises

Define the list *abstract data type*

Discuss lists in Python (keywords and how to use)

In-Class Exercise

Given an integer, print the following pattern for |i|=x rows:

On row i, print the numbers 1 through x-i

```
def numberTriangle(x):
```

```
1 2 3 4
1 2 3
1 2
1
```

In-Class Exercise

Given an integer, return the number of digits

```
def countDigits(x):
```

```
95718 = 5
  777 = 3
    0 = 1
```

List Abstract Data Type

What is a list? What properties does it have? What functions?

List Abstract Data Type

A list is an **ordered** collection of items

- Items can be either **heterogeneous** or **homogenous**
- The list can be of a **fixed size** or is **resizable**

List Abstract Data Type

A minimally functional list must have the following functions:

Constructor

Insert

Delete

Index

Size()**

In-Class Exercise

Using the ADT **[Constructor, insert, delete, index, size()]**, draft a function that will check if two lists are equal to each other.

```
def isEqual(list1, list2):
```

Building the foundation in CS 277

The most important skill in this class is being able to **use** data structures

Understanding **when** and **why** to use a particular structure comes next

Lists in Python: Constructor

There are many ways to construct (or initialize) a Python list.

```
l1 = [1, 2, 3]


l2 = list( (3, 4, 5) )


l3 = l1+l2


l4 = l1.copy()


print(l1 is l4)
```

Lists in Python: Insert

`Insert()` tries to add the object at a particular index in the list

`Append()` adds the item to the end of the list

```
l1 = []


l1.insert(2, “A")


l1.append(“C")


l1.insert(1, "B")


print(l1)
```

Lists in Python: Remove

`Remove()` removes the first instance of the object

`Remove()` crashes if the object doesn't exist!

`Pop()` removes and returns the last object in the list

```
l1 = [1,2,1,3,1,4]


l1.remove(7)


l1.remove(1)


print(l1)


x = l1.pop()


print(x, l1)
```

Lists in Python: Index

`Index()` returns the index of the first matching item in the list

`__getItem__(), []` returns the index of the first matching item in the list

```
l1 = [1,2,1,3,1,4]

print(l1.index(1))


print(l1.index(5))


print(l1.index(4))


print(l1[0])


print(l1[10])


print(l1[-1])
```

Lists in Python: Size

len() is not a class method but works for most forms of a list in Python

Remember: Python indexing starts at 0

```
l1 = [1,2,1,3,1,4]

print(len(l1))
```

List Abstract Data Type

A minimally functional list must have the following functions:

Constructor: `__init__()`

Insert: `append(x)` `insert(i, x)`

Delete: `remove(x)` `pop()`

Index `__getitem__() []` `index(x)`

Size()** `len(list)`

In-Class Exercise

What is the return value for this function?

```
def inclass():
    l = [1,2,3,4,5,6]

    x = l.pop()

    l.pop()

    l.pop()

    l.insert(0, x)

    l.append(8)

    l += [1, 2]

    return l
```

Multi-dimensional lists

7	8	9
4	5	6
1	2	3

Lists in Python store objects. Lists in Python are objects.

```
def makeMatrix():
```

```
M[2] =
```

```
M[1][0] =
```

```
M[0][2] =
```

In-Class Exercise

Lets finish writing tic-tac-toe!

Variables:

Gameboard

TurnTracker

Winner

Functions:

makeMove

checkWinner

Python List

There are many implementations of lists in Python. Here are three*:

```python
myList = [1, 2, 3, 4, 5]

print(myList)

print(len(myList))

print(myList[2])
```

```python
myTuple = (1, 2, 3, 4, 5)

print(myTuple)

print(len(myTuple))

print(myTuple[2])
```

```python
import numpy as np
myNP = np.array([1,2,3,4,5])

print(myNP)

print(len(myNP))

print(myNP[2])
```

Consider for next time

Why are there so many different implementations?