CS218 - Data Structures
FAST NUCES Peshawar Campus
Dr. Nauman (recluze.net)

August 26, 2019

1 Linked List in Python

Raster images of the notebook 04-linked-list

Linked List

In [17]:
```
class Node:
    def __init__(self, data=None):
        self.val = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None
```

The Push Operation

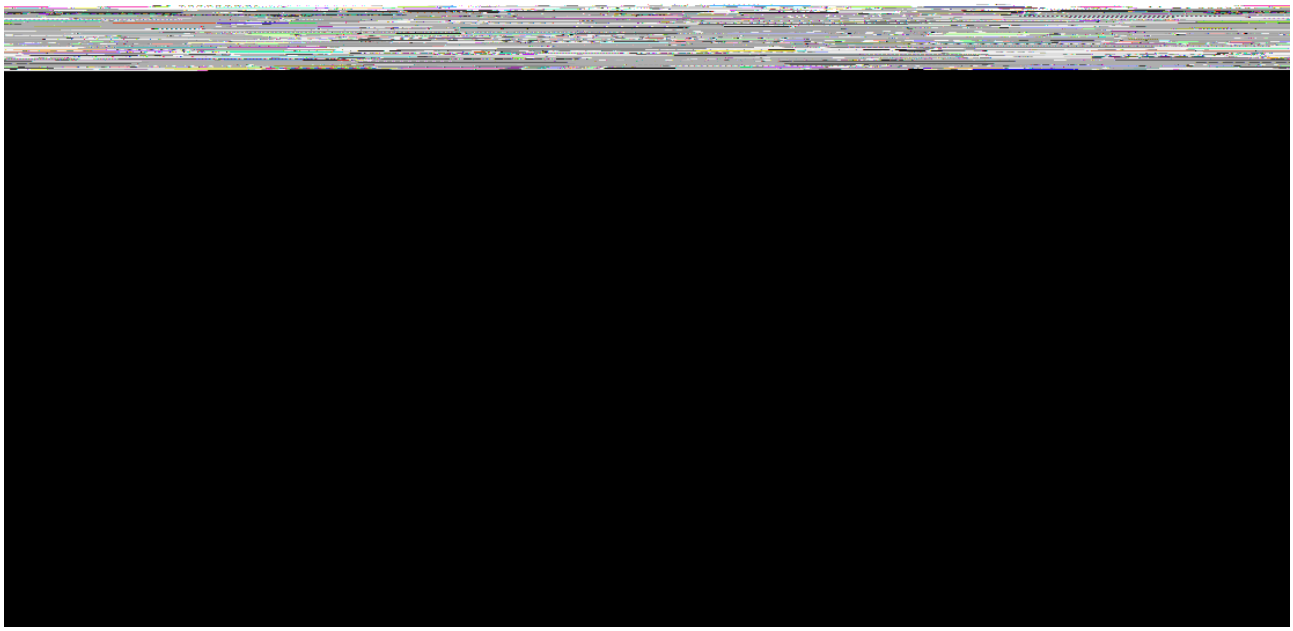
Push operation has two cases:

1. When there are no nodes
2. When there is already one or more nodes

In [18]:
```
def push(self, val):
    new_node = Node(val)

    # no node currently
    if self.head is None:
        self.head = new_node
        return

    # otherwise, reach the end and then insert
    last = self.head
    while last.next is not None:
        last = last.next

    last.next = new_node

LinkedList.push = push   ## We can add functions to classes even after definition
```

Insertion

Again, two cases:



In [25]:
```
l = LinkedList()
l.push(1)
l.push(2)
l.push(3)
l.insert(0, 10)
print(l)

l.insert(1, 11)
print(l)
l.insert(1000, 12)
print(l)
l.insert(5, 121)
print(l)
```

```
Case 1
[10, 1, 2, 3]
Case 2
[10, 11, 1, 2, 3]
Case 2
[10, 11, 1, 2, 3, 12]
Case 2
[10, 11, 1, 2, 3, 121, 12]
```

Remove Operation

This is also the same:

1. If first node is present and same as val, remove it.
2. Otherwise, move *prev* and *temp* until temp points to the value. Set next of *prev* to next of *temp*. (Temp is lost)

In [27]:
```
def remove(self, val):

    temp = self.head

    # check first node
    if temp is not None:
        if temp.val == val:
            print("case 1")
            self.head = temp.next
```

In [23]:
```
# Todo: len, get(index)
```