In [3]:
```
# Implementation of double link list
class Node(object):
    # Doubly linked node
    def __init__(self, data=None, next=None, prev=None):
        self.data = data
        self.next = next
        self.prev = prev

class doubly_linked_list(object):
    def __init__(self):
        self.head = None
        self.tail = None
        self.count = 0

    def append_item(self, data):
        # Append an item
        new_item = Node(data, None, None)
        if self.head is None:
            self.head = new_item
            self.tail = self.head
        else:
            new_item.prev = self.tail
            self.tail.next = new_item
            self.tail = new_item

        self.count += 1

    def print_foward(self):
        for node in self.iter():
            print(node)

    def iter(self):
        # Iterate the list
        current = self.head
        while current:
            item_val = current.data
            current = current.next
            yield item_val

items = doubly_linked_list()
items.append_item('PHP')
items.append_item('Python')
items.append_item('C#')
items.append_item('C++')
items.append_item('Java')

print("Items in the Doubly linked list: ")
items.print_foward()
```

```
Items in the Doubly linked list:
PHP
Python
C#
C++
Java
```

In [2]:
```
## Implementing linklist using collections.deque()

#importing module
import collections

#initialising a deque() of arbitary length
linked_lst = collections.deque()

#filling deque() with elements
linked_lst.append('first')
linked_lst.append('second')
linked_lst.append('third')

print("elements in the linked_list:")
print(linked_lst)

#adding element at an arbitary position
linked_lst.insert(1,'fourth')

print("elements in the linked_list:")
print(linked_lst)

#deleting the last element
linked_lst.pop()

print("elements in the linked_list:")
print(linked_lst)

#removing a specific element
linked_lst.remove('fourth')

print("elements in the linked_list:")
print(linked_lst)
```

```
elements in the linked_list:
deque(['first', 'second', 'third'])
elements in the linked_list:
deque(['first', 'fourth', 'second', 'third'])
elements in the linked_list:
deque(['first', 'fourth', 'second'])
elements in the linked_list:
deque(['first', 'second'])
```