

API Documentation

API Documentation

March 28, 2014

Contents

Contents	1
1 Package coinor.blimpy	2
1.1 Modules	2
1.2 Variables	2
2 Module coinor.blimpy.LinkedList'	3
2.1 Variables	3
2.2 Class LinkedList	3
2.2.1 Methods	3
2.2.2 Properties	5
2.3 Class Node	5
2.3.1 Methods	6
2.3.2 Properties	6
3 Module coinor.blimpy.Queues	7
3.1 Variables	7
3.2 Class Queue	7
3.2.1 Methods	8
3.2.2 Properties	8
3.3 Class PriorityQueue	8
3.3.1 Methods	9
3.3.2 Properties	10
4 Module coinor.blimpy.Stack'	11
4.1 Variables	11
4.2 Class Stack	11
4.2.1 Methods	11
4.2.2 Properties	12

1 Package coinor.blimpy

1.1 Modules

- **LinkedList'**: Lists Module A basic linked list implementation conforming to the Python list API.
(Section 2, p. 3)
- **Queues**: This module implements a Queue class, a simple list-based queue data structure and a PriorityQueue class, which is a heap-based priority queue data structure.
(Section 3, p. 7)
- **Stack'**: A basic stack implementation using a linked list.
(Section 4, p. 11)

1.2 Variables

Name	Description
<code>--package--</code>	Value: <code>'coinor.blimpy'</code>

2 Module *coinor.blimpy.LinkedList*

Lists Module A basic linked list implementation conforming to the Python list API. It can be used as a drop-in replacement for the built-in list class. Created on Jan 29, 2012

Version: 1.1.0

Author: Ted Ralphs, Aykut Bulut (ted@lehigh.edu, ayb211@lehigh.edu)

License: BSD

2.1 Variables

Name	Description
<code>--email--</code>	Value: 'ayb211@lehigh.edu'
<code>--maintainer--</code>	Value: 'Aykut Bulut'
<code>--package--</code>	Value: 'coinor.blimpy'
<code>--title--</code>	Value: 'Linked list data structure'
<code>--url--</code>	Value: None

2.2 Class *LinkedList*

object  `coinor.blimpy.LinkedList'.LinkedList`

implementation of link list data structure.

The behavior is designed to be the same as a Python list.

For efficiency when using the list as a stack, the list is stored such that the last item in the list is the head node. Thus, the append, push, pop, and most other methods are efficient, but forward iteration is not. Reverse iteration is efficient, however,
 pre: Node is the head node of a linked list and length is the length of that list

post: creates a *LinkedList* type object

2.2.1 Methods

<code>--add--(<i>self</i>, <i>otherLinkedList</i>)</code>

<code>--contains--(<i>self</i>, <i>item</i>)</code>

sequential search method pre: <i>self</i> , <i>item</i> to be searched post: True if list contains the item, False otherwise
--

<code>--delitem--(<i>self</i>, <i>position</i>)</code>
--

`__getitem__(self, index)`

replace built-in class method that returns the item for the given index
 pre: self, index, index should be less than length of list, list should not be empty
 post: return item for the given index

`__init__(self, Node=None, length=0)`

constructor method of the class pre: self
 Overrides: object.__init__

`__iter__(self)`

built-in class method, makes LinkedList objects iterable pre: self post: self.head, first Node on the list

`__len__(self)`

class method that returns the number of items in the list pre: self post: returns number of items in the list

`__repr__(self)`

repr(x)
 Overrides: object.__repr__ extit(inherited documentation)

`__reversed__(self)`

built-in class method, makes LinkedList objects reverse iterable pre: self post: self.head, first Node on the list

`append(self, item)`

class method that appends the given item at the end of the list pre: self, item

`backward(self)`

`count(self, item)`

class method that counts the number of occurrences of item in the list
 pre: self, item
 post: number of occurrences of item in the list

`extend(self, otherLinkedList)`

class method that extends the list by adding otherLinkedList at the end of the self pre: self, otherLinkedList

forward (<i>self</i>)

index (<i>self</i> , <i>item</i>)
--

class method that returns the index of the first occurrence of item, returns None if there is no any item. pre: self, item post: item index or None

insert (<i>self</i> , <i>position</i> , <i>item</i>)

class method that inserts item to the given position pre: self, position, item position should not be greater than length of the list

peek (<i>self</i> , <i>index</i> =None)

class method that retrieves, but does not remove, the head (first element) of this list pre: self post: the data of the head of the list or None if the list is empty

pop (<i>self</i> , <i>index</i> =None)
--

class method that removes the item at the given position in the list, and returns it. If no index is specified removes and returns the last item in the list pre: self, position (optional), position should be less than length of the list, list should not be empty post: return the item at given index or last item if not specified

remove (<i>self</i> , <i>item</i>)

class method that removes the first occurrence of the given item if there is one pre: self, item
--

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

2.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

2.3 Class Node

object —
coinor.blimpy.LinkedList'.Node

Basic data type that LinkedList will contain pre: data that node will contain post: Node type object

2.3.1 Methods

`__init__(self, initdata, nextNode=None)`

constructor of Node class pre: Node class object (self), initial data (initdata)

Overrides: object.__init__

`__repr__(self)`

repr(x)

Overrides: object.__repr__ extit(inherited documentation)

`__str__(self)`

str(x)

Overrides: object.__str__ extit(inherited documentation)

`getData(self)`

class method that returns to data pre: self post: data of the Node

`getNext(self)`

`setData(self, newdata)`

class method that sets data pre: self, new data

`setNext(self, newnext)`

class method that changes the next Node pre: self, new next

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`,
`__setattr__()`, `__sizeof__()`, `__subclasshook__()`

2.3.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

3 Module coinor.blimpy.Queues

This module implements a Queue class, a simple list-based queue data structure and a PriorityQueue class, which is a heap-based priority queue data structure.

Version: 1.1.0

Author: Aykut Bulut, Ted Ralphs (ayb211@lehigh.edu,ted@lehigh.edu)

License: BSD

3.1 Variables

Name	Description
<code>__maintainer__</code>	Value: 'Aykut Bulut'
<code>__email__</code>	Value: 'ayb211@lehigh.edu'
<code>__url__</code>	Value: None
<code>__title__</code>	Value: 'Queue data structure'
<code>__package__</code>	Value: 'coinor.blimpy'

3.2 Class Queue

object —
 coinor.blimpy.Queues.Queue

A queue data structure built on top of a linked list
 attributes:

 items: A list that holds objects in the queue
 type: LinkedList

methods:

 __init__(self): constructor of the class
 isEmpty(self): returns True if the queue instance is empty
 push(self,item): inserts item to the queue
 pop(self,item): removes first item in the queue if no item is
 specified removes the given item if item is
 specified
 size(self): returns the size of the queue

3.2.1 Methods

`__init__(self)``x.__init__(...)` initializes `x`; see `help(type(x))` for signatureOverrides: `object.__init__` `extit`(inherited documentation)**`isEmpty(self)`****`enqueue(self, item)`****`push(self, item)`****`dequeue(self, item=None)`****`remove(self, item=None)`****`pop(self, item=None)`****`peek(self, item=None)`****`size(self)`**

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

3.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

3.3 Class *PriorityQueue*



A priority queue based on a heap.

attributes:

<code>heap:</code>	A heap-ordered list that holds objects in the queue type: list
<code>entry_finder:</code>	A (map) dictionary for finding items in the heap
<code>counter:</code>	A unique sequence generator
<code>size:</code>	Number of items in the queue

methods:

<code>__init__(self):</code>	constructor of the class
<code>isEmpty(self):</code>	returns True if the queue is empty, False otherwise
<code>push(self, key, item, priority):</code>	inserts item with given key and priority into the queue
<code>pop(self, index):</code>	removes item with index from the queue if no item is specified or removes the given item if item is specified
<code>size(self):</code>	returns the size of the queue

3.3.1 Methods

`__init__(self, aList=None)`

`x.__init__(...)` initializes `x`; see `help(type(x))` for signature

Overrides: `object.__init__` `extit`(inherited documentation)

`isEmpty(self)`

`heapify(self)`

`pop(self, key=None)`

Remove and return the lowest priority task. Raise `KeyError` if empty.

`peek(self, key=None)`

`get_priority(self, key)`

`push(self, key, priority=None, item=None)`

Add to the heap or update the priority of an existing task.

remove (<i>self</i> , <i>key</i>)
--

Mark an existing task as REMOVED. Raise <code>KeyError</code> if not found.

Inherited from object

`--delattr--()`, `--format--()`, `--getattr--()`, `--hash--()`, `--new--()`, `--reduce--()`, `--reduce_ex--()`,
`--repr--()`, `--setattr--()`, `--sizeof--()`, `--str--()`, `--subclasshook--()`

3.3.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>--class--</code>	

4 Module `coinor.blimpy.Stack`

A basic stack implementation using a linked list.

Version: 1.1.0

Author: Aykut Bulut, Ted Ralphs (ayb211@lehigh.edu,ted@lehigh.edu)

License: BSD

4.1 Variables

Name	Description
<code>__email__</code>	Value: 'ayb211@lehigh.edu'
<code>__maintainer__</code>	Value: 'Aykut Bulut'
<code>__package__</code>	Value: 'coinor.blimpy'
<code>__title__</code>	Value: 'Stack data structure'
<code>__url__</code>	Value: None

4.2 Class Stack

object └─ **`coinor.blimpy.Stack.Stack`**

This stack class is built on top of a linked list data structure.

4.2.1 Methods

```
__init__(self)
x.__init__(...) initializes x; see help(type(x)) for signature
Overrides: object.__init__ extit(inherited documentation)
```

```
__repr__(self)
repr(x)
Overrides: object.__repr__ extit(inherited documentation)
```

```
__str__(self)
```

```
str(x)
```

```
Overrides: object.__str__ extit(inherited documentation)
```

```
isEmpty(self)
```

```
peek(self, item=None)
```

```
pop(self, item=None)
```

```
push(self, item)
```

```
remove(self, item)
```

```
size(self)
```

Inherited from object

```
__delattr__(), __format__(), __getattr__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(),  
__setattr__(), __sizeof__(), __subclasshook__()
```

4.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

Index

- coinor (*package*)
 - coinor.blimpy (*package*), 2
 - coinor.blimpy.LinkedList' (*module*), 3–6
 - coinor.blimpy.Queues (*module*), 7–10
 - coinor.blimpy.Stack' (*module*), 11–12