
NAME

PseudoHeap

SYNOPSIS

```
use PseudoHeap;

use PseudoHeap qw(:all);
```

DESCRIPTION

PseudoHeap class provides the following methods:

`new`, `AddKeyValuePair`, `AddKeyValuePairs`, `DeleteKey`, `DeleteKeys`, `DeleteMaxKey`, `DeleteMinKey`, `GetCurrentSize`, `GetKeyType`, `GetKeyValues`, `GetKeys`, `GetMaxKey`, `GetMaxSize`, `GetMinKey`, `GetSortedKeys`, `GetType`, `SetKeyType`, `SetMaxSize`, `SetType`, `StringifyPseudoHeap`

PseudoHeap is designed to support tracking of a specific number of largest or smallest key/value pairs with numeric or alphanumeric keys along with corresponding scalar or reference values.

Although PseudoHeap is conceptually similar to a heap, it lacks number of key properties of a traditional heap data structure: no concept of root, parent and child nodes; no ordering of keys in any particular order; no specific location greatest or smallest key.

The keys are simply stored in a hash with each key pointing to an array containing specified values. The min/max keys are updated during addition and deletion of key/value pairs; these can be retrieved by accessing corresponding hash.

Addition and deletion of key/value is also straightforward using hashes. However, min/max keys need to be identified which is done using Perl sort function on the keys.

FUNCTIONS

`new`

```
$NewPseudoHeap = new PseudoHeap(%NamesAndValues);
```

Using specified parameters *NamesAndValues* names and values hash, `new` method creates a new object and returns a reference to a newly created `NewPseudoHeap` object. By default, the following property names are initialized:

```
Type = undef;
KeyType = undef;
MaxSize = 10;
```

Examples:

```
$NewPseudoHeap = new PseudoHeap(
    'Type' => 'KeepTopN',
    'KeyType' => 'Numeric');

$NewPseudoHeap = new PseudoHeap(
    'Type' => 'KeepTopN',
    'KeyType' => 'AlphaNumeric',
    'MaxSize' => '20');

$NewPseudoHeap = new PseudoHeap(
    'Type' => 'KeepBottomN',
    'KeyType' => 'AlphaNumeric',
    'MaxSize' => '20');
```

`AddKeyValuePair`

```
$PseudoHeap->AddKeyValuePair($Key, $Value);
```

Add specified *Key* and *Value* pair to pseudo heap using a new or an existing key and returns PseudoHeap.

`AddKeyValuePairs`

```
$PseudoHeap->AddKeyValuePairs(@KeyValuePairs);
```

Adds multiple key and value pairs specified in array *KeyValuePairs* to pseudo heap using a new or existing keys and returns PseudoHeap.

DeleteKey

```
$PseudoHeap->DeleteKey($Key);
```

Deletes a specified *Key* from pseudo heap and returns PseudoHeap.

DeleteKeys

```
$PseudoHeap->DeleteKeys(@Keys);
```

Deletes a specified *Keys* from pseudo heap and returns PseudoHeap.

DeleteMaxKey

```
$PseudoHeap->DeleteMaxKey();
```

Deletes a *MaxKey* along with its associated values from pseudo heap and returns PseudoHeap.

DeleteMinKey

```
$PseudoHeap->DeleteMinKey();
```

Deletes a *MinKey* along with its associated values from pseudo heap and returns PseudoHeap.

GetCurrentSize

```
$Size = $PseudoHeap->GetCurrentSize();
```

Returns current *Size* of pseudo heap corresponding to number to keys in heap.

GetKeyType

```
$KeyType = $PseudoHeap->GetKeyType();
```

Returns *KeyType* of pseudo heap. Possible *KeyType* values: *Numeric* or *Alphanumeric*.

GetKeyValues

```
@Values = $PseudoHeap->GetKeyValues($Key);  
$NumOfValues = $PseudoHeap->GetKeyValues($Key);
```

Returns an array containing *Values* associated with a specified *Key* in pseudo heap. In scalar context, it returns number of values associated with a key.

GetKeys

```
@Keys = $PseudoHeap->GetKeys();  
$NumOfKeys = $PseudoHeap->GetKeys();
```

Returns an array containing all *Keys* in pseudo heap. In scalar context, it returns total number of keys.

GetMaxKey

```
$MaxKey = $PseudoHeap->GetMaxKey();
```

Returns *MaxKey* present in pseudo heap.

GetMaxSize

```
$MaxSize = $PseudoHeap->GetMaxSize();
```

Returns *MaxSize* of pseudo heap.

GetMinKey

```
$MinKey = $PseudoHeap->GetMinKey();
```

Returns *MinKey* present in pseudo heap.

GetSortedKeys

```
@Keys = $PseudoHeap->GetSortedKeys();
$NumOfKeys = $PseudoHeap->GetSortedKeys();
```

Returns an array containing all sorted Keys in pseudo heap. In scalar context, it retruns total number of keys.

Keys are sorted based on values of Type and KeyType for pseudo heap:

Type	KeyType	SortOrder	SortOperator
KeepTopN	Numeric	Descending	<=>
KeepTopN	Alphanumeric	Descending	cmp
KeepBottomN	Numeric	Ascending	<=>
KeepBottomN	Alphanumeric	Ascending	cmp

GetType

```
$Type = $PseudoHeap->GetType();
```

Returns *Type* of pseudo heap.

SetKeyType

```
$PseudoHeap->SetKeyType($KeyType);
```

Sets *KeyType* of pseudo heap and returns PseudoHeap.

SetMaxSize

```
$PseudoHeap->SetMaxSize($MaxSize);
```

Sets *MaxSize* of pseudo heap and returns PseudoHeap.

SetType

```
$PseudoHeap->SetType($Type);
```

Sets *Type* of pseudo heap and returns PseudoHeap.

StringifyPseudoHeap

```
$PseudoHeapString = $PseudoHeap->StringifyPseudoHeap();
```

Returns a string containing information about *PseudoHeap* object

AUTHOR

Manish Sud <msud@san.rr.com>

COPYRIGHT

Copyright (C) 2025 Manish Sud. All rights reserved.

This file is part of MayaChemTools.

MayaChemTools is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.