Heap Class Documentation

Source File: Utils.h
Namespace: aa
Class Header: `template <class T> class Heap : public Object`

Overview

The *Heap* class is a heap array class for a generic type.

Constructors

- `Heap()` (default constructor)
  - **Purpose:** Allocates and initializes a dynamic generic array to size 20 with the default value of the generic type and assigns 0 to the heap's size.
- `Heap(const Heap<T>& obj)` (copy constructor)
  - **Purpose:** Constructs a deep copy of *obj*.
  - **Parameter(s):**
    - *obj*: Constant *Heap* reference object.
- `Heap(size_t sz)`
  - **Purpose:** Allocates and initializes a dynamic generic array to size *sz* and the default value of the generic type and assigns 0 to the heap's size. However, if *sz* is 0, the array is allocated to 20.
  - **Parameter(s):**
    - *sz*: A possible size for the generic dynamic array.
- `Heap(initializer_list<T> lst)`
  - **Purpose:** Allocates and initializes a dynamic generic array to the size and elements of *lst* and assigns 0 to the heap's size.
  - **Parameter(s):**
    - *lst*: A list of elements of the generic type.

Destructor

- `~Heap()`
  - **Purpose:** Deallocates the generic array.

Assignment Operators

- `operator=(const Heap<T>& rhs)`
  - **Purpose:** Constructs a deep copy of *rhs*.
  - **Parameter(s):**
    - *rhs*: Constant *Array* reference object.
  - **Return:** `*this`.
- `operator=(initializer_list<T> lst)`
  - **Purpose:** Allocates and initializes a dynamic generic array to the size and elements of *lst*, and assigns 0 to the heap's size.
  - **Parameter(s):**
    - *lst*: A list of elements of the generic type.
  - **Return:** `*this`.

Methods

- `size() const`
  - **Purpose:** Gets the heap's size.
  - **Return:** The capacity of heap.
- `length() const`
  - **Purpose:** Gets the capacity of the generic dynamic array.
  - **Return:** The capacity of the generic array.

- `size(size_t sz)`
  - **Purpose:** Sets the size of the heap to *sz* if *sz* does not exceed the array's capacity.
  - **Parameter(s):**
    - *sz*: A possible size of the heap.
- `heapView(bool val)`
  - **Purpose:** Sets the view of the data structure to the heap dataset (*val* = true) or the array (*val* = false).
  - **Parameter(s):**
    - *val*: A view switch.
- `operator[](unsigned int idx) const`
  `operator[](unsigned int idx)`
  - **Purpose:** Retrieves an element of the generic dynamic array or heap dataset with the index *idx*.
  - **Parameter(s):**
    - *idx*: A possible index of the generic dynamic array.
  - **Return:** A (constant) reference of an element of the generic dynamic array or heap dataset.
  - **Exception(s):**
    - `Out-Of-Bound Error`: Thrown if *idx* exceeds or equals the capacity of the generic dynamic array or heap [based on the view].
  - `toString() const override`
    - **Purpose:** Provides a string representation of the *Heap* object.
    - **Return:** A string representation of the elements of the generic dynamic array or the heap dataset, all enclosed within square braces.