**TADs**

|  |  |  |  |
| --- | --- | --- | --- |
| **PrirorityQueue TAD** | | | |
| PriorityQueue = { <E1 « p1 >, <E2 « p2 >, <…> , <En « pn > } | | | |
| { inv: pi ≥ pj , ∀ i ≥ j } | | | |
| Primitive operations: | | | |
| * PriorityQueue |  | * PriorityQueue | Constructor |
| * isEmpty | PriorityQueue | * Boolean | Analyzer |
| * enqueue | PriorityQueue X Element X Priority | * PriorityQueue | Modifier |
| * dequeue | PriorityQueue | * Element | Modifier |
| * peek | PriorityQueue | * Element | Analyzer |
| * size | PriorityQueue | * Integer | Analyzer |
| * getHeap | PriorityQueue | * List | Analyzer |
| * remove | PriorityQueue X Element | * PriorityQueue | Modifier |
| * clone | PriorityQueue | * PriorityQueue | Constructor |

|  |
| --- |
| **PriorityQueue** |
| Constructor of the Queue class that creates an empty queue. { pre: TRUE}  { post: *priorityQueue* = { heap = NIL } |

|  |
| --- |
| **isEmpty(***priorityQueue***)** |
| Checks if the priority queue is empty.  { pre: TRUE }  { post: True if *heap* = NIL  False if *heap* != NIL } } |

|  |
| --- |
| **enqueue(***priorityQueue, item, priority***)** |
| Adds an item with a specified priority to the priority queue.  { pre: TRUE }  { post: priorityQueue = { <E1 « p1 >, <E2 « p2 >, <…>, <Ek « pk > , <…> , <En « pn > } ,  pk  ≥ priority ≥ pk+1 } |

|  |
| --- |
| **dequeue(***priorityQueue***)** |
| Removes and returns the item with the highest priority from the priority queue  { pre: n > 0 }  { post: dequeue = <E1 « p1 >, priorityQueue = { <E2 « p2 >, <…>, <Ek « pk > } } |

|  |
| --- |
| **peek(***priorityQueue***)** |
| Returns the item with the highest priority without removing it from the priority queue.  { pre: TRUE }  { post: dequeue = <E1 « p1 > } |

|  |
| --- |
| **size(***priorityQueue***)** |
| Returns the current number of items in the priority queue.  { pre: TRUE }  { post: size = n } |

|  |
| --- |
| **getHeap(***priorityQueue***)** |
| Returns a reference to the internal heap data structure used in the priority queue. This method provides access to the heap for debugging and other purposes.  { pre: True }  { post: heap = { <E2>, <…> , <En> } } |

|  |
| --- |
| **remove(***priorityQueue, elem***)** |
| Removes an item from the priority queue based on its value.  { pre: priorityQueue = { <E1 « p1 >, <E2 « p2 >, <…>, <Ek « pk = elem> , <…> , <En « pn > } }  { post: priorityQueue = { <E1 « p1 >, <E2 « p2 >, <…> , <En « pn > } } |

|  |
| --- |
| **clone(***priorityQueue***)** |
| Overrides the clone method to create a deep copy of the PriorityQueue.  { pre: TRUE }  { post: priorityQueue.clone } |