|  |  |  |
| --- | --- | --- |
| MINHEAP | | |
| MinHeap= {Heap= <T1, T2,…,Tn>} | | |
| {Inv: ∀i, T1<Ti } | | |
| MinHeap:  insert:  peekMin:  isEmpty:  addElements:  extractMin:  getHeap: | MINHEAP x T  MINHEAP  MINHEAP  MINHEAP x ARRAYLIST<T>  MINHEAP  MINHEAP | 🡪 MINHEAP  → MINHEAP  🡪 T  🡪 BOOLEAN  → MINHEAP  → T  🡪 ARRAYLIST<T> |

**Constructor Operations:**

|  |
| --- |
| **MinHeap()**  “Creates a new MinHeap”  {pre: TRUE}  {post: MinHeap= {Heap= <ArrayList<T>>} |

**Modifying Operations:**

|  |
| --- |
| **insert(Heap, T)**  “Inserts a new element into the heap”  {pre: MinHeap={Heap= <T1, T2,…,Tn>}, T=<content>}  {post: MinHeap={Heap= <T1, T2,…,Tn,Tn+1> }} |

|  |
| --- |
| **addElements(MaxHeap, ArrayList<T>)**  “Adds a new group of elements to the MinHeap (with the certainty that the process maintains the invariant of the structure)”  {pre: MinHeap={Heap= < T1, T2,…,Tn>} , Elements={< R1, R2,…,Rn>} }  {post: MinHeap={Heap= < T1, T2,Ti,Tj,…,Tn , R1, R2,…,Rn >} |

|  |
| --- |
| **extractMin(Heap)**  “Returns the smallest element (the first one, knowing that the structure is always following its invariant) and removes it from the heap”  {pre: MinHeap={Heap=< T1, T2,…,Tn>} }  {post: T1=<content>} |

**Analyzing Operation:**

|  |
| --- |
| **peekMin()**  “Returns the smallest element (the first one, knowing that the structure is always following its invariant) of the heap”  {pre: MinHeap={Heap= < T1, T2,…,Tn>} }  {post: T1=<content>} |

|  |
| --- |
| **isEmpty()**  “Returns whether the heap structure has any elements or not”  {pre: MinHeap={Heap= < ArrayList<T>>} }  {post: TRUE if Heap contains no elements, else FALSE} |

|  |
| --- |
| **getHeap(Heap)**  “Returns a list with all the elements contained in Heap (with the elements organized following the invariant)”  {pre: MinHeap={Heap= < T1, T2,…,Tn>} }  {post: Heap= < T1, T2,…,Tn> } |