|  |
| --- |
| TAD Action Class |
| Action = {Action Type = <actionType>, Task Details = <taskDetails>, ID = <id>} |
| {inv: Action.actionType = String, Action.taskDetails = Task, Action.id = int} |
| Primitive Operations:  getActionType (): -> Action Class  setActionType (String actionType): actionType -> Action Class  getTaskDetails (): -> Action Class  setTaskDetails (Task taskDetails): taskDetails -> Action Class  getId (): -> Action Class  setId (int id): id -> Action Class |

|  |
| --- |
| getActionType ():  \*Returns the action’s type |
| (Pre: TRUE)  (Post: Action U {actionType}) |

|  |
| --- |
| setActionType (String actionType):  \*Stablishes a new action type |
| (Pre: newActionType)  (Post: TRUE) |

|  |
| --- |
| getTaskDetails ():  \*Returns the task’s details |
| (Pre: TRUE)  (Post: Action U {taskDetails}) |

|  |
| --- |
| setTaskDetails (Task taskDetails):  \*Stablishes a new task detail |
| (Pre: newTaskDetails)  (Post: TRUE) |

|  |
| --- |
| getId ():  \*Returns the action’s ID |
| (Pre: TRUE)  (Post: Action U {id}) |

|  |
| --- |
| setId (String id):  \*Stablishes a new ID |
| (Pre: newID)  (Post: TRUE) |

|  |
| --- |
| TAD Task Class |
| Task = {Title = <title>, Description = <description>, Date Limit = <dateLimit>,  Priority = <priority>} |
| {inv: Task.title = String, Task.description = String, Task.dateLimit = String, Task.priority = Priority.PRIORITY V Priority.NO\_PRIORITY} |
| Primitive Operations:  getTitle (): -> Task Class  setTitle (String title): title -> Task Class  getDescription (): -> Task Class  setDescription (String description): description -> Task Class  getDateLimit (): -> Task Class  setDateLimit (String dateLimit): dateLimit -> Task Class  getPriority (): -> Task Class  setPriority (Priority priority): priority -> Task Class |

|  |
| --- |
| getTitle ():  \*Returns the task’s title |
| (Pre: TRUE)  (Post: Action U {actionType}) |

|  |
| --- |
| setTitle (String title):  \*Stablishes a new task title |
| (Pre: newTaskTitle)  (Post: TRUE) |

|  |
| --- |
| getDescription ():  \*Returns the task’s description |
| (Pre: TRUE)  (Post: Task U {description}) |

|  |
| --- |
| setDescription (String description):  \*Stablishes a new task description |
| (Pre: newTaskDescription)  (Post: TRUE) |

|  |
| --- |
| getDateLimit ():  \*Returns the task’s date limit |
| (Pre: TRUE)  (Post: Task U {dateLimit}) |

|  |
| --- |
| setDateLimit (String dateLimit):  \*Stablishes a new task date limit |
| (Pre: newTaskDateLimit)  (Post: TRUE) |

|  |
| --- |
| getPriority ():  \*Returns the task’s priority |
| (Pre: TRUE)  (Post: Task U {priority}) |

|  |
| --- |
| setPriority (Priority priority):  \*Stablishes a new task pirority |
| (Pre:newTaskPriority)  (Post: TRUE) |

|  |
| --- |
| TAD Stacks |
| Stack = {Elements = <elements>} |
| {inv: Stack.elements = List of Action} |
| Primitive Operations:  push(Action action): action -> Stack  pop(): -> Action  peek(): -> Action  isEmpty(): -> Boolean |

|  |
| --- |
| push(Action action):  \*Adds an action to the top of the stack |
| (Pre: action ≠ null)  (Post: Stack U {action}) |

|  |
| --- |
| pop():  \*Removes and returns the action at the top of the stack |
| (Pre: Stack ≠ empty)  (Post: Action) |

|  |
| --- |
| peek():  \*Returns the action at the top of the stack without removing it |
| (Pre: Stack ≠ empty)  (Post: Action) |

|  |
| --- |
| isEmpty():  \*Checks if the stack is empty |
| (Pre: TRUE)  (Post: Stack = empty) |

|  |
| --- |
| TAD Priority Queue |
| PriorityQueue = {Elements = <elements>} |
| {inv: PriorityQueue.elements = List of Pair(Task, Priority)} |
| Primitive Operations:  enqueue(Task task, Priority priority): task, priority -> PriorityQueue  dequeue(): -> Task  peek(): -> Task  isEmpty(): -> Boolean |

|  |
| --- |
| enqueue(Task task, Priority priority)  \*Adds a task to the queue according to its priority |
| (Pre: task ≠ null, priority ≠ null)  (Post: PriorityQueue U {task}) |

|  |
| --- |
| dequeue()  \*Removes and returns the task with the highest priority from the queue |
| (Pre: PriorityQueue ≠ empty)  (Post: Task) |

|  |
| --- |
| peek()  \*Returns the task with the highest priority from the queue without removing it |
| (Pre: PriorityQueue ≠ empty)  (Post: Task) |

|  |
| --- |
| isEmpty()  \*Checks if the queue is empty |
| (Pre: TRUE)  (Post: PriorityQueue = empty) |

|  |
| --- |
| TAD Hash Table |
| HashTable = {Tasks = <tasks>} |
| {inv: Task.Key != Task.otherKey } |
| Primitive Operations  values():     -> HashTable  clear():       -> HashTable  put():          -> HashTable  get(id):       -> HashTable  remove(id):  -> HashTable |

|  |
| --- |
| values()  \*Returns a list of all the values stored in the hash table |
| (Pre: TRUE)  (Post: HashTable ) |

|  |
| --- |
| clear()  \*Removes all the elements from the hash table |
| (Pre: TRUE)  (Post: HashTable = { ∅}) |

|  |
| --- |
| put()  \*Updates or adds a new value in the hash table |
| (Pre: Task)  (Post: ) |

|  |
| --- |
| get(id)  \*Returns the value |
| (Pre:Task.Key)  (Post: Task(id)} |

|  |
| --- |
| remove(id)  \*Removes the first occurrence of a specified element |
| (Pre:Task.Key)  (Post: ) |