**Integrative Task#1 Computation and Discrete Structures 1**

**By: Juan Pablo Parra, Stick Martinez and Daniel Gonzalez**

**Specification of requirements:**

**SOFTWARE ENGINEERING PROBLEM SPECIFICATION TABLE**

|  |  |
| --- | --- |
| CLIENT | Owner of the task and reminder system |
| USER | Any person using the system |
| FUNCTIONAL REQUIREMENTS | **RF1**: Store tasks and reminders  **RF2:** Add tasks  **RF3:** Modify tasks  **RF4:** Delete tasks  **RF5:** Undo method |
| CONTEXT OF THE PROBLEM | Creating a task and reminder management system entails three core functions: task and reminder addition, modify, and delete. To effectively capture and store both elements, we must also incorporate a system for prioritization, classifying tasks into two main categories: "Priority" and "Non-priority." Priority tasks should be registered in the task queue with higher visibility, ensuring that the most critical tasks are displayed prominently. Meanwhile, non-priority tasks are managed on a first-come, first-served basis. Additionally, it is crucial to implement an action logging feature to enable users to undo any actions they've taken, whether it involves adding, modifying, or deleting tasks or reminders. |
| NON-FUNCTIONAL REQUIREMENTS | RNF : Interfaz de usuario concreta fácil de comprender |

| **Name or identifier** | **RF1**: Store tasks and reminders | | |
| --- | --- | --- | --- |
| **Summary** | The program must be able to store tasks and reminders using a key that in our case is a unique identifier and the value is the information that is associated (object task) it means the title, description, deadline, priority. | | |
| **Entries** | Input name | Data type | Selection or repetition condition |
| Key | String | Must be unique |
| Value | String | Objecto : Title, description, deadline, priority |
| **Result or postcondition** | The task or reminder is stored by using an identifier | | |
| **Outputs** | Output name | Data type | Selection or repetition condition |
| ItStore | boolean | if true it was stored correctly, if false it was not stored correctly |

| **Name or identifier** | **RF2:** Add tasks | | | |
| --- | --- | --- | --- | --- |
| **Summary** | The program must be able to create a task which requires the following: register a title, description, deadline and its priority. Depending on the type of task to be registered, i.e. whether it is a priority or non-priority task, an additional structure (heap and queue respectively) is created to store it. | | | |
| **Entries** | Input name |  | Data type | Selection or repetition condition |
| Title |  | String | no more than 30 characters |
| Description |  | String | no more the 300 characters |
| Deadline |  | Calendar | Date |
| Priority |  | Int | PRIORITY 1,2,3  NOT\_PRIORITY 0 |
| **Result or postcondition** | Adds task to hash table and giving priority to the corresponding list. | | | |
| **Outputs** | Output name |  | Data type | Selection or repetition condition |
| taskAdded |  | boolean | if true it was stored correctly, if false it was not stored correctly |

| **Name or identifier** | **RF3:** Modify tasks | | |
| --- | --- | --- | --- |
| **Summary** | The program must allow the user to modify all of the attributes stored in the task. Depending on which one you want to change the program will allow it without changing the other attributes | | |
| **Entries** | Input name | Data type | Selection or repetition condition |
| Title | String | no more than 30 characters |
| Description | String | Less than 300 characters |
| DeadLine | Calendar | Date |
| Priority | Int | PRIORITY 1,2,3  NOT\_PRIORITY 0 |
| **Result or postcondition** | Modify the task. If the priority is changed from a priority to a non-priority, it should be removed from the priority list and added to the non-priority list and vice versa. | | |
| **Outputs** | Output name | Data type | Selection or repetition condition |
| taskModified | boolean | if true it was modified correctly, if false it was not modified correctly |

| **Name or identifier** | **RF4:** Delete tasks | | |
| --- | --- | --- | --- |
| **Summary** | The program must allow the user to delete a task. It should ask whether you want to delete the priority list or the non-priority list. Deleting a task must remove it from the storage and the corresponding list. | | |
| **Entries** | Input name | Data type | Selection or repetition condition |
| WhichTask | Int | PRIORITY 1  NOT\_PRIORITY 2 |
| **Result or postcondition** |  | | |
| **Outputs** | Output name | Data type | Selection or repetition condition |
| deleteTask | boolean | if true it was deleted correctly, if false it was not deleted correctly |

| **Name or identifier** | **RF5:** Undo method | | |
| --- | --- | --- | --- |
| **Summary** | The program must allow a method that relives the immediate action made by the user. That is, if a task is deleted and the undo method is applied, the task must appear in the system. | | |
| **Entries** | Input name | Data type | Selection or repetition condition |
| **Result or postcondition** | Reverts the immediate accition. | | |
| **Outputs** | Output name | Data type | Selection or repetition condition |
| msg | String | (if there are no more shares to be revert) |