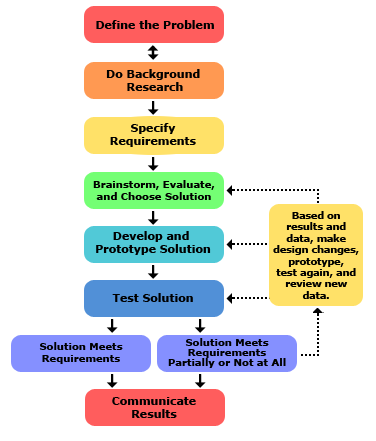
**Método de la ingeniería**



**Problem identification**

**REQUIREMENTS IDENTIFICATION**

The health care institution requires a system to manage the admission to its laboratory units, specifically two. A correct management requires the administration of the clients and their corresponding data, taking into account that the order of the queue (priority) depends on some data. The solution must comply with the requirements stipulated by the healthcare institution. Taking into account that the system will be used by some personnel of the institution, the system must be intuitive and with the least possible number of failures.

**Problem definition**

**Functional Requirements:**

**R1:** The system must be able to search for a patient arriving at the laboratory through his or her ID, obtaining all other patient data, since the priority of care depends on some data.

**R2:**The system must allow the registration of a patient not registered in the system. To register the patient, some data must be requested, such as name, age, identification document, gender, if there is any important base disease, if the patient is a woman, she is asked if she is pregnant and the date of admission to the laboratory is recorded.

**R2.1:** The system must allow to update the date of the last visit of a client.

**R.3** The system must allow assigning a priority to each patient according to the client's data recorded in the database.

**R4**:According to this priority the patient is assigned to the laboratory admission queue, in addition to taking into account the order of arrival. If the patient does not have any priority, he/she enters the queue with general access.

**R5:** The system must allow the software user (institution personnel) to undo both the admission and discharge of a patient in any of the available laboratories.

**R6:** The system should allow monitoring the list of people in the laboratory, their priority of care, and the action to discharge each patient to continue with the next one.

**Process requirements:**

**RP1:** The construction of the software must be done in groups of 3 people. (in our case 4 :P)

**Non-Functional Requirements:**

**RNF 1:** The system must be easy to use for the people at the reception of the clinic where the software will be used.

**RNF2:** The system must be secure to store all patient information in the database with a secure method

**2. Collection of information**

**Important base diseases:** These types of diseases constitute a very important group because they occur frequently and also because they can affect the intellectual or physical quality of the person who suffers it.

**Source: scielo** [**https://scielo.isciii.es/scielo.php?script=sci\_arttext&pid=S1137-66272008000400008#:~:text=Las%20enfermedades%20de%20base%20gen%C3%A9tica,tipo%20de%20problemas%20que%20producen**](https://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1137-66272008000400008#:~:text=Las%20enfermedades%20de%20base%20gen%C3%A9tica,tipo%20de%20problemas%20que%20producen)**.**

**Egress:** Egress refers to anything that leaves a particular place or space.

Source: DefinicionABC

https://definicionabc.com/egreso/

**Entry:** Entry of a person or thing into a group, set or institution**.**

**Source:** Oxford Languages

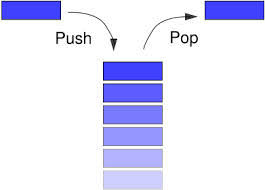
**Prioritization format:** ehCos proposes a prioritization software which works through questions asked to the patient and hierarchies, depending on the patient's answers the system will put him/her in one level of prioritization or in another.

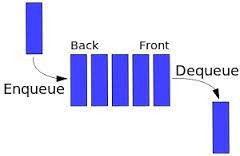
**Shift systems**: Shift systems allow to control the flow of people in a given space, these systems are supported by a highly compatible hardware, its operation is commonly based on the user takes a turn in a machine and the software stores it with date and time, when the shifts that are ahead have passed the system will indicate the next shift.

**3. Search of creative solutions**

**AlternativE 1. Queue and Stack.**

.

**Stack Queue**



In this alternative, the problem can be solved by making use of the structures called stacks and queues. As the users who wish to enter the laboratories can be a large number and not all of them can enter at the same time, it is necessary to have an order in some kind of queue, that is why the queue structure will be useful because its methods enqueue and dequeue represent the entry of a new person to the queue and the exit of a person to the laboratory respectively. For the functionality of requirement R4 in this alternative, we can use a stack that works together with the patient queue, every time a patient is discharged to the laboratory, this will be stacked with the stacking method of this type of structures, in this way the staff using the software will be able to undo the discharge with the unstacking method and the reference of that person who entered the laboratory will not be lost.

**Stack:**

the stack represents a lineal data structure of data in wich we can add or eliminate elements only for one of the ends.in consequence the elements of a stack are eliminated in the inverse order that they be add. thanks to this characteristics is called LIFO(last input, first output).

.

**Example:**

Stacks with linear structures such as arrays. since their components occupy successive places in the DE and each has only one successor/predecessor, except for the first/last.

**Queue:**

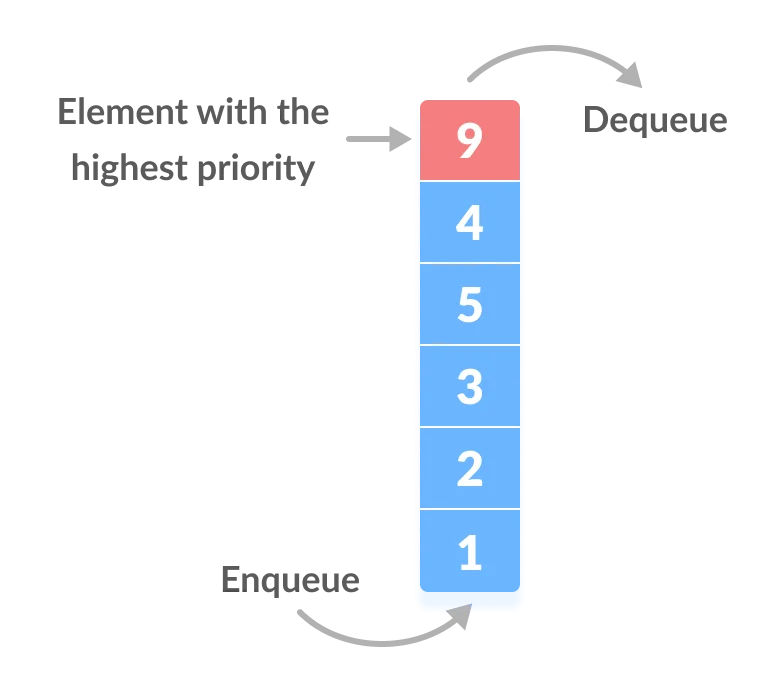
A queue is a data structure in which the access mode to its elements is FIFO (First Input First Output). It allows the storage and retrieval of data, that is to say the insertion and removal of elements from the queue according to the FIFO principle. When an item is added to the queue it is added last. When an item is removed from the queue the one at the front of the queue, that is the first is removed.

**Alternative 2.Pencil and paper.**

In this alternative, the problem is approached from an unorthodox point of view, but it also provides a solution to the problem. The management of admissions to the health care institution can be done with the use of some type of folio and by hand by the different laboratory managers. As in the previous alternative, two folios or as many as necessary can be used to manage both the queue of patients and those who have already been admitted for care, although compliance with requirement R4 would be complicated to implement in this alternative, but not impossible.



**Alternative 3. Priority queue and stacky.**

****

**Priority queue:**

A priority queue is a set of elements in which each of these is associated with a sorting value called priority, this type of queue is characterized by admitting insertions with new elements, in addition to the consultation and elimination of the element which is associated with the lowest priority.

That is to say, the ordering of the queue will be defined by the priority of each of its elements, and the higher the priority, the higher it will be at the top of the queue.

**4. Transition from Ideas to Preliminary Designs**

**Stacks and queues:** While this alternative may be a good choice for the project because we can have many users who want to enter and can not enter all at once. so we need to make a queue and for this we will use the queue structure. because when someone wants to enter the queue we will glue and when someone leaves we will unglue and the stack structure will be used when for example a patient is discharged it will be applied and if the person who manages the software wants to undo the discharge can unstack and the reference of the person who entered the laboratory will not be lost.

However with this alternative we have a big problem and it is that in our environment there will be patients that have different diseases or special characteristics, these patients will have a priority over the rest of users so we need a way to prioritize them is at this point where we decided that this alternative should be discarded.

**Pencil and paper:** This alternative can solve all the project requirements, however it is not efficient at all, it can generate delays and confusion.

**Priority Stack and queues:** This alternative is like the first one but in this one the priority problem is solved, so at the moment of the development of the project this can be the one that favors us the most for the solution of the project.

**5. Evaluation and selection of the best solution.**

**The criteria for evaluating and selecting the best solution will be:**

**Criterion A Efficiency.** A solution with better efficiency than the others considered is preferred. The efficiency can be:

[3] Very efficient

[2] Efficient

[1] Not very efficient

**Criterion B Complexity**. The least confusing and easiest for a user to handle would be preferred. Complexity can be:

[3]not very complex

[2]complex

[1]Very complex

**Criterion C**. The alternative that cannot help to develop all the requirements of the project without exception will be preferred. The rank will be:

[2]Develops all requirements.

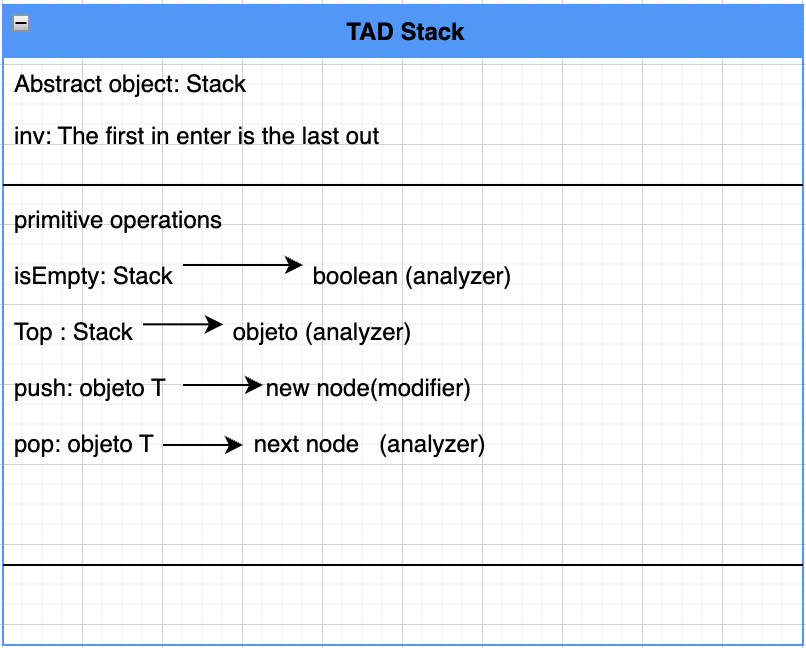
[1]Develops most of the requirements, however presents problems in some of them

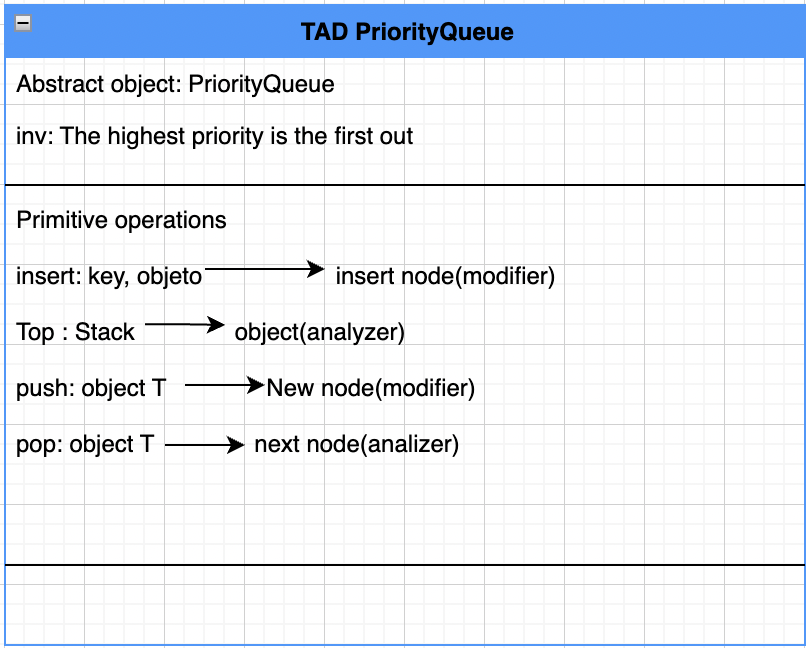
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **criteria A** | **criteria B** | **criteria C** | **Total** |
| **Alternative 1:**  **stack and queue** | **3** | **2** | **1** | **6** |
| **Alternative 2:**  **paper and pencil** | **1** | **1** | **1** | **3** |
| **Alternative 3:**  **Priority Stack and queues** | **3** | **3** | **2** | **8** |

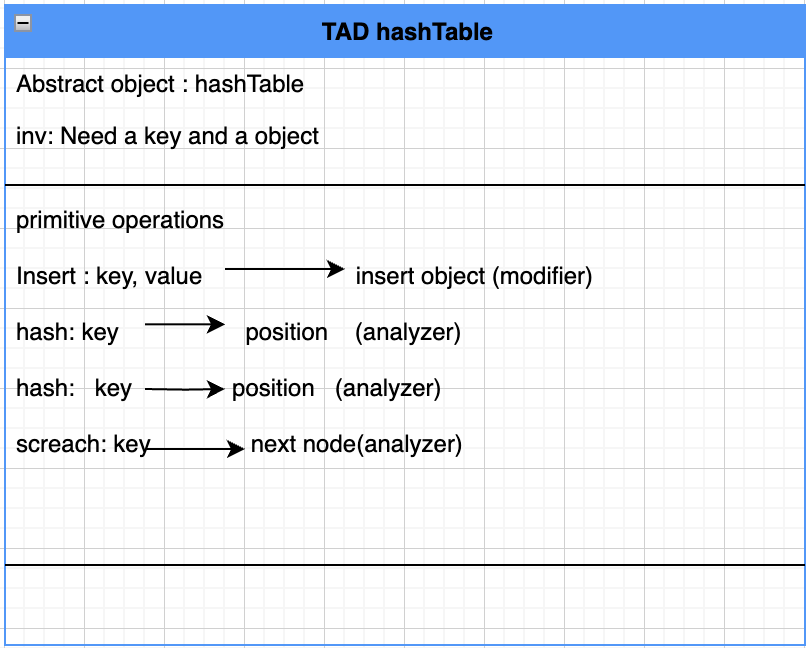
According to the previous evaluation, the alternative to be used is the 3rd alternative, since it was the one that obtained the highest score and therefore is the one that best meets our criteria and project.

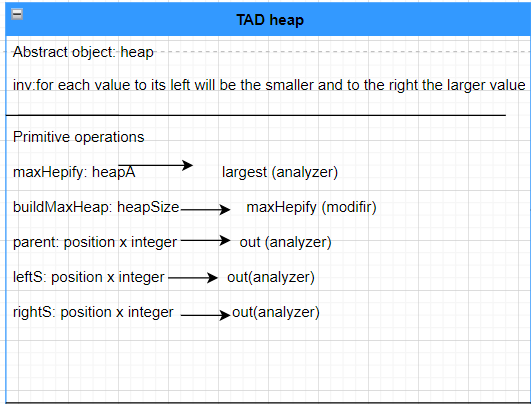
**Report preparation and specification:**

**TAD:**

****

****

****

****