**Engineering Method**

1. IDENTIFICATION OF THE PROBLEM:

Create the first version of a system that allow the management of the entrance and the exit of the patients in a clinical laboratory.

The system must load the information of a flat text file into the data base of the IPS for use in the program

Once a patient passes his id information in the reception, the system must search in the most efficient way the information of the patient in the data base and show it. Else if the patient is not in the data base, he should be registered.

The system must assign a certain priority to each patient.

The system must delegate each patient to the Hematology or general unit.

Once a patient gets to a unit, he should be assigned to one of the two type of attention: Priority access and general access depending on the priority he has.

The system must allow the user to undo the last entry or exit action he just performed. Deleting the patient’s entrace or exit.

The system must have an option to monitor the number of patients inside the laboratory and the attention order in each unit’s queue.

The system must have at least two different type of data structures.

The data structures implemented in the system must be created generically.

1. RECOPILATION OF THE NECESSARY INFORMATION:

It is now necessary for us to search information regarding all the concepts related to the problem. In this way we would be able to gather ideas of how we could later on create the solution for the problem.

**DEFINITIONS:**

Generics in Java:

The generic programming in Java allow us to create methods, classes and interfaces with parameterized types, which makes possible for these entities to work with different data types.

With this type of programming we can use the dynamic classes on Java such as ArrayList, HashMap, Queue, etc. and create them in a generic way.

Generic method:

Generic Java method takes a parameter and returns some value after performing a task. It is exactly like a normal function; however, a generic method has type parameters that are cited by actual type. This allows the generic method to be used in a more general way. The compiler takes care of the type of safety which enables programmers to code easily since they do not have to perform long, individual type castings

Hashmap:

The Hashtable class implements a hash table, which stores key/value pair in hash table. Any non-null object can be used as a key or as a value. To successfully store and retrieve objects from a hashtable, the objects used as keys must implement the hashCode method and the equals method.

PriorityQueue:

Priority Queue is an abstract data type that is similar to a queue, and every element has some priority value associated with it. The priority of the elements in a priority queue determines the order in which elements are served (i.e., the order in which they are removed). If in any case the elements have same priority, they are served as per their ordering in the queue.

Queue:

A queue is defined as a linear data structure that is open at both ends and the operations are performed in First In First Out (FIFO) order.

We define a queue to be a list in which all additions to the list are made at one end, and all deletions from the list are made at the other end.  The element which is first pushed into the order, the operation is first performed on that.

ArrayList:

ArrayList provides us with dynamic arrays in Java. It can be helpful in programs where lots of manipulation in the array is needed. The main advantages of ArrayList are, if we declare an array then it’s needed to mention the size but in ArrayList, it is not needed to mention the size of ArrayList if you want to mention the size then you can do it.

Array:

An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array).

Linked List:

A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers as shown in the below image.

Java Swing:

Swing is a Java Foundation Classes library and an extension of the Abstract Window Toolkit. Swing offers much-improved functionality over AWT, new components, expanded components features, excellent event handling with drag and drop support.

Java FX:

JavaFX is a Java library and a GUI toolkit designed to develop and facilitate Rich Internet applications, web applications, and desktop applications. The most significant perk of using JavaFX is that the applications written using this library can run on multiple operating systems and several. This characteristic makes it very versatile across operating systems and different platforms.

1. Search for creative solutions:

* Brain Storm:

GUI ALTERNATIVES:

ALTERNATIVE 1 JAVA SWING:

Java Swing allow us to use simple codes, which does not alter our usual way of programming.

ALTERNATIVE 2 JAVA FX:

Java FX require us to use the scene builder app to create the different scenes, relations and functionalities of each method in each scene, as they could be the buttons or the blanks of text.

QUEUE ATTENTION ALTERNATIVES:

ALTERNATIVE 1 PRIORITY QUEUE & QUEUE:

We could create both a Priority Queue and a Queue to attend the patients in the different units they are. All the patients that possess a any type of priority would be added to the Priority Queue, where they would be attended base on the amount of priority they have on a descending order. While all the other patients that do not possess any type of priority would be added to the Queue, where they would be attended in order of arrival.

ALTERNATIVE 2 DOUBLE PRIORITY QUEUE:

We could also create two Priority Queues which would be divided into priority and general access. All the patients that possess a any type of priority would be added to the Priority Queue of priority, where they would be attended base on the amount of priority they have on a descending order. While all the other patients that do not possess any type of priority would be added to the Priority Queue of general access, where they would be attended in order of arrival.

ALTERNATIVE 3 PRIORITY QUEUE:

We could also create just one Priority Queue where we would add all the patients, but we would attend first the ones with any type of priority on a descending order and later on all the patients without any type of priority in order of arrival.

DATA BASE ALTERNATIVES:

ALTERNATIVE 1 HASHTABLE:

A hashtable would allow us to save the ids of the patients as their keys and all of their personal and medical information on the value slot. Also, while using a linked list in it we can avoid any collision.

ALTERNATIVE 2 ARRAYLIST:

An Arraylist would allow us to save all the information of the object patient in each slot.

# OF PATIENTS ON THE LAB ALTERNATIVES:

ALTERNATIVE 1 ARRAYLIST:

An ArrayList would allow us to store all the patients that have entered the lab and delete them every time we need to, not worrying on the number of patients we save in it.

ALTERNATIVE 2 ARRAY:

An Array would allow us to store all the patients that have entered the lab and delete them every time we need to.

1. **Transition from ideas to preliminary designs**:

In this step its important discard some ideas previously mentioned that are no workable to develop a solution to the problem, therefore, we need to set a correct structure of design problem.

First of all, we can discard the alternative 2 of the GUI, this due to the fact that it would take us too much time to understand and know how to apply correctly all the functionalities Java Fx has. And as we have to invest time also on implementing the program this alternative is not meant for us.

Then, we also decided to discard the alternative 2 of the data bases, because it is ask in the document that we need to search the information of the patients in the most efficient way possible, and this ArrayList would take too long when there were too much patients in the data base.

Finally, we decided to discard the alternative 2 in the # of patients on the lab, because we would need to define the size of the Array, which is not practical.

Now we Will focus on the advantages of the remaining alternatives:

DATA BASE:

ALTERNATIVE 1 HASHTABLE:

* Is really efficient while searching for data
* Allow us to dave the ids of the patients to distinct one form other
* Even if there could be a case of collision, the implemented linked list would solve it

QUEUES:

ALTERNATIVE 1 PRIORITY QUEUE & QUEUE:

* Both structures allow us to simulate a queue
* The queue, allow us to attend in arrival order
* The Priority queue, allow us to attend base on the priority each patient have

ALTERNATIVE 2 DOUBLE PRIORITY QUEUE:

* Solo hay que implementar una sola estructura
* We just need to implement one structure
* The patients would be attended in arrival order and base on the priority each have

ALTERNATIVE 3 PRIORITY QUEUE:

* We save all the patients in just one queue, which would be easier to store them

GUI:

ALTERNATIVE 1 JAVA SWING:

* Easy to use and implement
* Can be use in all alternatives

# PACTIENTS IN LAB ALTERNATIVES:

ALTERNATIVE 1 ARRAYLIST:

* Undefined size

1. **EVALUATION AND SELECTION OF THE PREFERRED SOLUTION:**

After having analyzed each alternative in the previous point, we can draw our preferred solutions:

Database:

• The HashTable is an excellent choice to use as a database because we can save and access the patient's information in a very simple way, since the Id can be used as the key, which will give us access to the information that this have.

Rows:

• Regarding queues, we conclude that the best solution is to use a double priority queue, since it is a structure that can also be used as a queue because by adding patients always with the same priority, they will come out in order. Of income. The above means that the priority queue can work in a similar way to the queue. Using the double queue also saves a lot of time because only one structure needs to be implemented, instead of implementing the queue and the priority queue separately.

User interface

• Regarding the user interface, we came to the conclusion that the best option is javax.swing since, despite not being the prettiest to look at, it is an interface that is easy to implement and helps us add certain functionalities. to the program that can provide us with many things when developing the code.