**Estructuras de Datos**

**Memoria PECL1 2021-2022**

**Integrantes del grupo**

Marco González Martínez 09064995M

Silvana Della Casa Tovar 48619234V

***Detalles y justificación de la implementación***

***Especificación de los TAD’s implementados, definición de las operaciones.***

**TAD Pacientes**

espec PACIENTES

usa booleanos, naturales

genero Paciente

operaciones

0: -> paciente {genera datos del paciente}

parcial esHernia: ->bool

parcial getID: paciente -> natural

fespec

**TAD Cola**

espec COLA[PACIENTES]

usa natural

parámetro

genero pacientes

genero cola

operaciones

[ ]: ->cola {genera cola vacia}

insertar: paciente cola -> cola

parcial extraer: cola -> cola

parcial primero: cola -> paciente

parcial getLongitud: cola -> natural

var c: cola; v: paciente; n: natural <-0

ecuaciones

extraer (insertar (v, c)) = c

primero (insertar (v, c) = v

func getLongitud c: cola dev n: natural

si (insertar (v, c))

n<- n+1

finsi

finfunc

**TAD Pila**

espec PILA[PACIENTES]

usa natural

parámetro

genero pacientes

genero pila

operaciones

[ ]: ->pila {genera pila vacia}

insertar: paciente pila -> pila

parcial extraer: pila -> pila

parcial cima: pila -> paciente

parcial getLongitud: pila -> natural

var p: pila; v: paciente; n: natural <-0

ecuaciones

extraer (insertar (v, p)) = p

cima (insertar (v, p)) = v

func getLongitud p: pila dev n:natural

si insertar (v, p)

n<-n+1

finsi

finfunc

**TAD Lista**

espec LISTA[PACIENTES]

usa natural

parámetro

genero pacientes

genero lista

operaciones

[ ]: -> lista

insertar: paciente lista -> lista

parcial extraer: lista-> lista

parcial cabeza: lista -> paciente

parcial tail: lista -> paciente

parcial getLongitud: lista -> natural

var l: lista; v: paciente; n: natural <-0

ecuaciones

extraer (v:l) = l

cabeza (v:l) = v

tail (v#l) = v

func getLongitud l: lista dev n:natural

si insertar (v, p)

n<-n+1

finsi

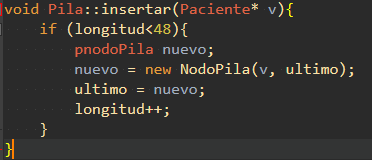
fincfunc

***Dificultades encontradas: Soluciones adoptadas***

Una dificultad encontrada fue a la hora de crear el método insertar en la clase lista debido a que los pacientes debían estar ordenados en las listas, y al contrario que en pilas y colas las listas admiten valores no solo al principio o al final, sino también entremedias. Por esto los pacientes tendrían que entrar en lista ordenados y directamente en su lugar correspondiente. Para ello usa la clase NodoLista creando punteros nuevo y aux. Nuevo toma como valor el ID del paciente a insertar, comparando con cada uno de los IDs de pacientes ya en lista representados por aux.

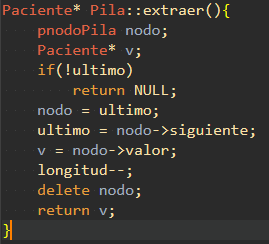
***Métodos más destacados***

**Clase Pila: insertar**

****

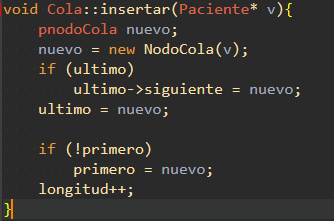
Usando punteros de clase NodoPila, creamos un puntero nuevo que usaremos para añadir el paciente v a la pila siempre que la longitud de esta no sea ya de 48 pacientes para no exceder el limite. El puntero ultimo pasa a representar el nuevo paciente, y aumentamos longitud.

**Clase Pila: extraer**

****

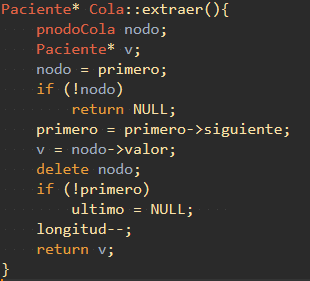
Dado que es una pila el valor a extraer será el ultimo añadido. Para ello creamos un puntero nodoPila nodo y un paciente vacío. Si la pila no es vacía (hay un valor ultimo), nodo representa el ultimo, y ultimo pasa a ser el siguiente en pila. V toma los datos del paciente en nodo, reducimos longitud, eliminamos nodo y devolvemos el paciente v.

**Clase Cola: insertar**

****

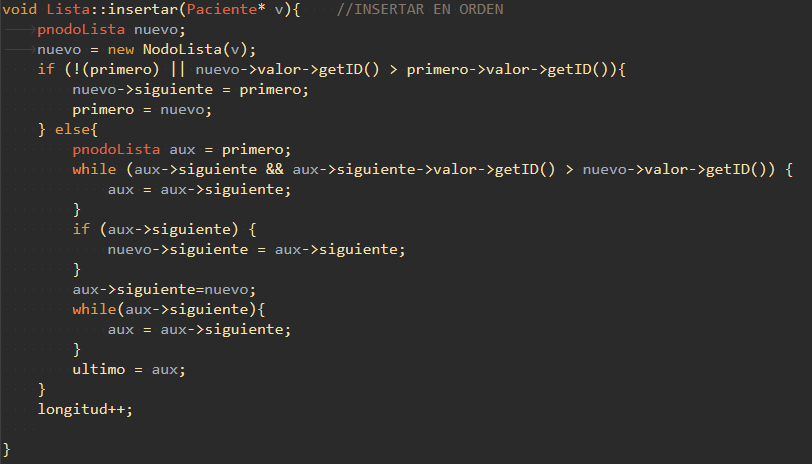
Para este método empleamos la clase NodoCola para crear la cola mediante punteros. Ultimo, primero, siguiente y nuevo son todos de clase NodoCola. Si el valor primero existe, es decir la cola no es vacía, va al puntero ultimo. Aquí se le asigna el paciente nuevo al valor siguiente en la cola y queda así como el último de la cola. Una vez insertado se aumenta el valor de longitud de la cola.

**Clase Cola: extraer**



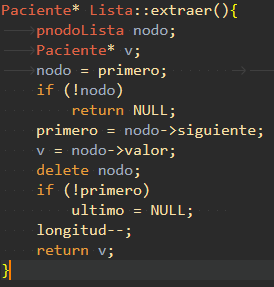
De nuevo empleamos la clase NodoCola para usar punteros a lista. Como es una cola, tendremos que extraer la cabeza de la cola. Para ello creamos un puntero nodo que tome como valor el primero y un paciente sin datos v. Si no existe la cola es vacía por lo que no devuelve nada. Si no, el puntero primero pasa a tomar como valor el siguiente paciente de la cola, y v toma como datos los del paciente al que apunta nodo, y elimina nodo, si este era el único elemento en cola daremos a ultimo valor nulo. Por último reducimos la longitud y devolvemos como valor el paciente extraído v.

**Clase Lista: insertar**



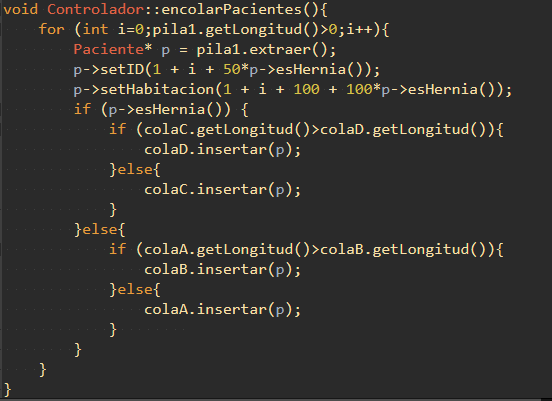
Como se ha comentado más arriba, para este método se usa la clase NodoLista creando punteros nuevo y aux. Nuevo toma el valor de ID del paciente que se quiere añadir, que va comparando con cada ID ya en lista hasta encontrar un ID mayor (aux toma los valores de cada uno de ellos y cambiando al siguiente cuando es menor). Al encontrar un ID mayor nuevo toma el lugar del paciente que en ese momento era el siguiente en la cola y siguiente pasa a ser el paciente ya en lista con el que se estaba comparando, quedando así ordenado. Por ultimo se aumenta la longitud de la cola.

**Clase Lista: extraer**



Mediante punteros NodoLista, creamos un nuevo puntero nodo y un paciente vacío v. Ya que la lista estaba ordenada por gravedad, el paciente a extraer será siempre el primero por lo que en efecto este método será muy similar al extraer en Cola, descrito previamente. Al igual que en cola, nodo toma el valor de primero siempre que la lista tenga datos, y primero pasará a hacer referencia al siguiente de la lista. V será el paciente extraído por nodo, que se elimina. Si no quedan más pacientes último será nulo, devolvemos el paciente v y reducimos longitud.

**Clase controlador: encolarPacientes**

****

Este método se encarga de sacar pacientes de la pila y colocarlos en las colas con menor número de pacientes según su enfermedad. Este método además asigna a los pacientes un ID (1-49 para apendicitis, 50-99 para hernia) y una habitación (101-199 apendicitis, 201-299 hernia). La forma en que hace esto es al desapilar pacientes toma el bool esHernia como valor 0 o 1, usándolo para añadir o no 50 al id inicial y 100 a la habitación para así conseguir el rango de valores deseado para cada enfermedad. Usa este bool también para comprobar en que cola deber ir e insertándola en la de menor pacientes comparando longitudes de ambas.

***Comportamiento del programa***

Al iniciar el programa nos encontraremos con el menú de opciones del hospital. Para iniciar la simulación primero deberemos crear datos de la clase paciente (hasta un máximo de 48) mediante la opción A que serán añadidos a una pila directamente. Para cada paciente se creará un bool representante de la enfermedad que tiene, hernia o apendicitis.

A partir de aquí podremos o consultar los datos de este pila o borrarlos, pero para continuar la simulación usaremos la opción D para simular la llegada de estos a las salas del hospital, representadas por la clase cola, de la cual habrá cuatro, dos por enfermedad. El paso de los pacientes a las colas se hará desapilando los pacientes uno a unos y añadiéndolos a las colas A y B de apendicitis, o C y D de hernias. Los pacientes se añadirán a la cola que tenga menos pacientes en ese momento.

Una vez hecho esto podremos de nuevo consultar los datos de estas colas o borrarlos. Para seguir pasamos a la fase de preoperatorio mediante la opción H, en la cual los pacientes salen de las colas y son añadidos a quirófanos representados por la clase lista. Tenemos dos litas, una para hernia y otra para apendicitis. A este punto se le asigna a cada paciente un ID y un número de habitación, ambos dependientes de la enfermedad y el ID será menor cuanta mayor prioridad tenga el paciente. Por esto los pacientes se desencolan y se añaden a las listas no solo en función de la enfermedad, sino que entraran a ellas de forma ya ordenada dependiendo de la prioridad de su ID.

Con esto concluye la simulación del hospital, las opciones que nos quedan aquí serán consultar los datos de pacientes en lista, mostrar los datos de el paciente de menor prioridad de apendicitis y mayor de hernia, o reiniciar el programa a su estado inicial borrando todos los datos de pacientes, pila, colas y listas.

***Bibliografía***

Hemos utilizado como apoyo los apuntes que figuran en el aula virtual de Blackboard facilitados por el profesor.