ESTRUCTURA DE DADES PRÀCTICA 2

Anàlisi de requeriments

Els principals requisits per a dur a terme aquesta pràctica son principalment 3:

-tenir una sèrie de certs coneixements de Java i l'ús d'un programa per a implementar aquest codi.

-dispondre d'un editor de text per a realitzar l'informe posterior tal com el Word i passar-lo després a PDF.

-tenir els recursos suficients per superar tots el problemes que et trobis durant la realització de la practica, com ara molta paciència i ganes de treballar.

Estudi Previ

Per a anar iniciant el desenvolupament de la practica primerament tractarem la implementació dels diferents TADs que empraré, per fer això, crearé una classe que serà el TAD equip que heretarà d'una interfície i per altra banda implementaré una interfície competició que hem servirà per donar herència a tres de les classes que conformen la practica, la classe Dinàmica, Estàtica i Fantasma, que son les tres implementacions de llista que se'ns demana.

Per anar finalitzant implementaré una classe que serà filla del TAD de equips, la classe ordenada i posteriorment generaré totes les classes necessàries per controlar les excepcions definides. Per altra banda, dintre de les classes Dinàmica, Estàtica i Fantasma genero una altra classe per a implementació del node i els seus punters o cursors, segons el cas.

Últimament crearé el main on ajuntaré totes les funcions descrites al llarg de les classes per a generar finalment un programa 100% funcional.

Decisions preses durant el desenvolupament de la pràctica

-Per a controlar que el programa no deixi de funcionar durant el seu funcionament, he decidit crear cinc excepcions diferents, una que per controlar que no hi dos jugador a la llista de jugadors amb el mateix identificador, una altra per controlar quan no existeix un jugador a la llista, una altra per comprovar quan la llista esta plena i no hi caben mes jugadors o equips, una altra per controlar que no hagi dos equips a la llista amb el mateix nom, una altra per controlar que la llista no estigui buida i finalment una altra per controlar quan no existeix un equip a la llista. També he decidit crear un mètode anomenat llegirenter() per tal de controlar que el valor introduït sigui un enter on es demana un numero, evitant així molts errors.

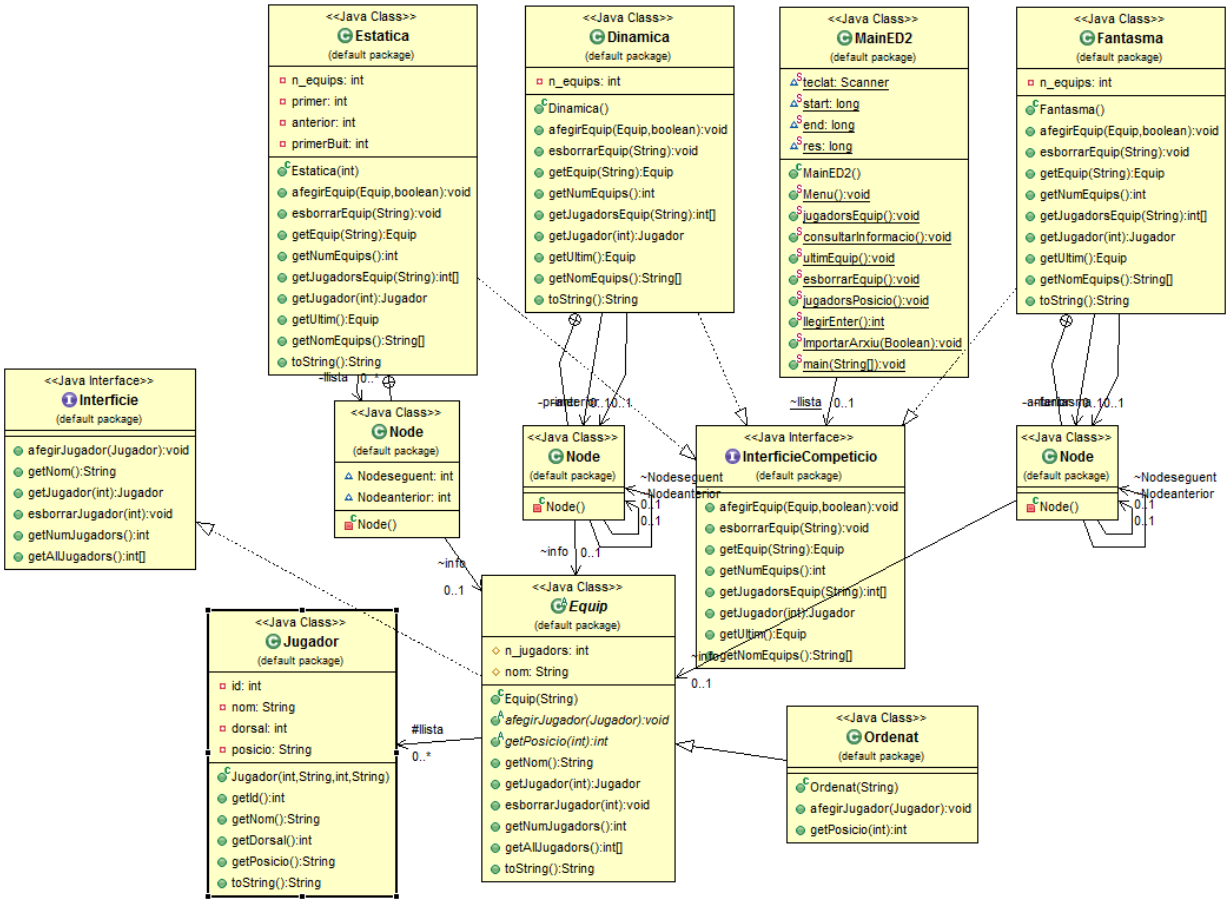
-Tal com ens ha especificat el professor a les classes de practiques, per tal de fer el programa el mes eficient possible, a la filla del TAD equip nomes he definit 2 funcions diferents, ja que son els que canviaran segons si implementem una llista ordenada o desordenada, un mètode per afegir jugadors de forma ordenada, i un mètode per trobar la posició a la llista d'un jugador segons l'identificador de forma dicotòmica.

-Per a la implementació del mètode afegir equip(Dinàmica, estàtica o fantasma) podem observar 3 casos diferents pel mètode de afegir ordenat, si el node s'ha de posicionar el primer

de tots, enmig de dos altres elements o al final de tot, cadascun d'aquest possibles casos es força idèntic els uns amb els altres però cada un d'ells te una part exclusiva com per exemple si agrego el primer li haure de dir que el node anterior a aquest serà null o si agrego el últim li haure de dir que el node següent a aquest es null o si l'agrego enmig de dos elements actualitzar els punters dels elements ens conseqüència per aconseguir variar l'ordre dels elements, tampoc ens hem de oblidar de comprovar la ja existència del equip a la llista mitjançant un bucle.

-Per a la implementació de la llista de forma estàtica, he afegit una variable que apunta sempre al primer element buit (i també una llista de nodes) per a controlar els espais buits al agregar o eliminar un equip, cada cop que agregui o elimini un element de la llista actualitzo la variable primerBuit, ja que així aquesta sempre apuntarà al primer espai buit i allí serà on agregaré sempre el següent element, amb aquesta variable aconsegueixo que paregui que estigui treballant amb dos llistes diferents, una de nodes i una de espais lliures, quan en realitat nomes estem treballant amb una única llista.

Diagrama de classes utilitzades



Cost temporal de les operacions

COSTOS TEMPORALS	DINAMICA	ESTATICA	ELEMENT FANTASMA
afegirEquip	n	n	n
esborrarEquip	n	n	n
getNumEquips	1	1	1
getEquip	n	n	n
getUltim	1	1	1
getJugadorsEquip	N^2	N^2	N^2
getNomEquips	n	n	n
getJugador	$N*\log_2 n$	$N*\log_2 n$	$N*\log_2 n$

Aspectes millorables

Els principals aspectes a millorar suposo que seria intentar disminuir el codi que he implementat segons els mètodes, però ara per ara no disposo d'uns coneixements tan desenvolupats de java per reduir l'ocupació del codi mantenint la mateixa eficiència.

Un altre aspecte a millorar a part de disminuir l'ocupació del codi seria intentar fer mètodes encara mes eficients amb uns costos temporals mínims.

Per finalitzar també es podria implementar una interfície gràfica per no tenir que treballar mitjançant la consola tot el temps.