Exemple de Prova d'Avaluació Continuada

PRIMERA PART

Es vol dissenyar una estructura de dades per emmagatzemar un conjunt de campionats del popular joc "Apalabrados". En aquesta PAC farem una prova pilot d'una part del sistema (un conjunt reduït de funcionalitats) i treballarem amb volums d'informació petits per tal de que us familiaritzeu amb el problema a resoldre i "practiqueu" una mica el disseny i la composició d'estructures de dades per donar forma i solució al problema plantejat. Concretament, per a la realització de l'exercici considereu:

1 El nombre de campionats C que guardarem a la prova pilot és variable i relativament petit, d'uns centenars.

2 El nombre de partides P d'un campionat també és variable i relativament petit, d'uns centenars.

3 El nombre de paraules PA que formaran part del diccionari de paraules vàlides d'un campionat serà variable i relativament petit.

4 El nombre de lletres diferents disponibles L serà de 29, cada lletra té una puntuació associada en funció de lo comú que sigui (per exemple, la 'A' donarà 1 punt, la 'H' donarà 8 punts, etc).

5 El nombre de jugadors d'una partida JP serà de 2.

6 El nombre de fitxes disponibles F inicialment en una partida serà fix, 100. No confongueu les lletres disponibles a un campionat amb les fitxes d'una partida, les fitxes de les partides són la suma de totes les ocurrències de lletres disponibles per jugar. Tingueu en compte també que hi ha una proporció més elevada de fitxes per les lletres més comuns (per exemple, 12 fitxes de la lletra 'E', 2 fitxes de la lletra 'B', etc...). Podeu trobar la relació de fitxes i lletres en el següent enllaç: http://es.wikipedia.org/wiki/Scrabble

7 El tauler de cada partida tindrà 15 files i 15 columnes. Considereu que no hi ha caselles especials.

8 Cada jugador tindrà un màxim de 7 lletres per fer una jugada.

9 El nombre de jugades GP d'una partida és variable i relativament petit.

Exercici 1

Especifiqueu un TAD *Apalabrados* per guardar els campionats que permeti:

- Crear un campionat nou, el campionat tindrà un nom i aquest serà únic.
- Afegir una paraula al diccionari de paraules que es farà servir a tot el campionat.
- Afegir una lletra amb la seva puntuació i el nombre de fitxes d'aquesta lletra que hi haurà disponibles. Aquesta associació es farà servir a tot el campionat.
- Crear una nova partida, la partida tindrà un nom que la identificarà i, en el moment de crear-la, s'especificarà els dos jugadors que la jugaran i el campionat en el que participen.
- Afegir una fitxa a la partida d'un campionat. No es poden afegir fitxes a una partida començada ni més fitxes d'una lletra que les màximes indicades (a la mateixa lletra).
- Començar la partida. Inicialitza una partida d'un campionat i assigna a cada jugador les seves fitxes inicials. El repartiment de fitxes consisteix en agafar una fitxa associada a la partida (l'última inserida) per cada jugador de manera alternativa fins completar les seves 7 fitxes.
- Fer una jugada a una partida. Caldrà especificar el jugador que fa la jugada, la fila i columna on comença la paraula, l'orientació (horitzontal –d'esquerra a dreta- o vertical –de dalt a baix-) i les lletres necessàries per fer la paraula. Cal comprovar que la paraula sigui vàlida, que el jugador té totes les fitxes necessàries per fer-la, calcular la puntuació de la jugada i actualitzar les lletres disponibles del jugador i la seva puntuació. Considereu també que en aquesta primera versió els jugadors no poden canviar fitxes i que per la puntuació de la jugada només cal comptar la paraula especificada i no les possibles paraules addicionals generades a partir d'aquesta. NOTA: veure els aclariments al final d'aquest document.
- Esbrinar quantes vegades s'ha utilitzat una determinada paraula en el campionat.
- Esbrinar quina paraula ha obtingut la puntuació més alta al campionat.

Apartat a)

Dóna la signatura del TAD *Apalabrados*. És a dir, indica el nom que donaries a les operacions encarregades de cada funcionalitat requerida. Indica, també, quins caldria que fossin els paràmetres d'entrada i quina la sortida en cas que es necessités.

Apartat b)

Fes l'especificació contractual de les operacions del TAD *Apalabrados*. En la redacció de l'especificació pots fer servir, si et cal, qualsevol de les operacions del TAD. Pren com a model l'especificació de l'apartat 1.2.3 del Mòdul 1 dels materials docents. Es valorarà especialment la concisió (absència d'elements redundants o innecessaris), precisió (definició correcta del resultat de les operacions), completesa (consideració de tots els casos possibles en què es pot executar cada operació) i manca d'ambigüitats (coneixement exacte de com es comporta cada operació en tots els casos possibles) de la solució. És important respondre aquest apartat usant una descripció condicional i no procedimental. L'experiència ens demostra que no sempre resulta fàcil distingir entre les dues descripcions, és per això que fem especial èmfasi insistint que poseu molta atenció a les vostres definicions.

A títol d'exemple indicarem que la descripció condicional (la correcta a utilitzar en el contracte) d'omplir un got buit amb aigua seria:

> @pre el got es troba buit.
>
> @post el got és ple d'aigua.

En canvi una descripció procedimental (incorrecta per utilitzar en el contracte) tindria una forma semblant a:

> @pre el got hauria de trobar-se buit, si no es trobés buit s'hauria de buidar.
>
> @post s'acosta el got a l'aixeta i s'hi tira aigua fins que estigui ple.

Cal també tenir en compte que un contracte hauria de disposar d'invariant sempre que aquesta fos necessària per descriure el TAD.

Exercici 2

A l'Exercici 1 heu definit l'especificació d'un nou TAD, el TAD *Apalabrados*, ara us demanem que feu el disseny de les estructures de dades que formaran aquest TAD. Dissenyeu, doncs, el sistema per tal que sigui el màxim d'eficient possible, tant a nivell d'eficiència espaial com temporal, tenint en compte els volums d'informació i les restriccions especificades a l'enunciat.

Tingueu en compte només les operacions que es demanen a l'enunciat a l'hora de fer aquest disseny.

Apartat a)

Dubtem entre utilitzar un vector, una llista encadenada o una llista encadenada ordenada per a emmagatzemar les jugades. Justifiqueu quina creieu que és la millor opció.

Apartat b)

Dubtem entre utilitzar un vector, una cua, una pila o una llista encadenada per a emmagatzemar les fitxes. Justifiqueu quina creieu que és la millor opció.

Apartat c)

Feu un dibuix de l'estructura de dades global pel TAD *Apalabrados* on es vegin clarament les estructures de dades que trieu per representar cada una de les parts i les relacions entre elles. Cal que feu el dibuix de l'estructura complerta, amb totes els estructures que us permetin implementar les operacions definides a l'especificació.

Apartat d)

Justifiqueu totes les estructures de dades que heu triat a l'hora de fer el disseny del TAD. La justificació de cada una de les estructures de dades ha de ser de l'estil:

"Per guardar XXX triem una llista encadenada ordenada ja que el número d'elements no és gaire gran, ens fa falta accés directe i ens calen recorreguts ordenats."

Exercici 3

A l'Exercici 1 heu definit l'especificació del TAD *Apalabrados* amb les seves operacions i a l'Exercici 2 heu triat les estructures de dades per cada part del TAD. En aquest exercici us demanem que us fixeu en els algorismes que us serviran per implementar algunes de les operacions especificades i en l'estudi d'eficiència de les mateixes. Tingueu en compte que la implementació de les operacions va estretament lligat a l'elecció de les estructures de dades que hagueu fet.

Apartat a)

Feu el pseudocodi i l'estudi d'eficiència de l'operació que hagueu definit per fer una jugada a una partida. Per fer-ho heu de descriure breument el seu comportament indicant els passos que la componen (amb frases com ara: "inserir en l'arbre AVL / esborrar de la taula de dispersió / consulta del piló / ordenar el vector..."), dient l'eficiència asimptòtica de cada pas i donant l'eficiència total de l'operació.

Apartat b)

Feu el pseudocodi i l'estudi d'eficiència de l'operació que hagueu definit per esbrinar quina paraula ha obtingut la puntuació més alta al campionat. Igual que a l'exercici anterior, heu de descriure breument el seu comportament indicant els passos que la componen (amb frases com ara: "inserir en l'arbre AVL / esborrar de la taula de dispersió / consulta del piló / ordenar el vector..."), dient l'eficiència asimptòtica de cada pas i donant l'eficiència total de l'operació.

Apartat c)

Suposeu ara que volem oferir una funcionalitat que permeti reproduir alguna de les partides del campionat, és a dir, tornar a fer totes les jugades de les que es composa una partida. Canviaríeu alguna de les estructures de dades per tal que aquesta operació fos eficient? En cas afirmatiu, expliqueu quins canvis faríeu i per què.

Exercici 4

Indica quins dels TADs de la biblioteca de TADs de l'assignatura et semblen més adients per utilitzar-los en la implementació de cada una de les estructures de dades definides pel TAD *Apalabrados*.

Aclariment: Suposant que el tauler es troba en la situació indicada a l'esquerra, el jugador pot jugar "BROU" des de la posició 2,3 en vertical, però només necessita les lletres "B" i "O", a l'aprofitar les lletres "R" i "U" que ja es troben al tauler. És a dir, cal comprovar que "BROU" (la paraula que juga el jugador) és una paraula vàlida i que el jugador té les lletres "B" i "O" (les que necessita per completar la paraula que vol jugar).

		R	O	S	
				A	
I	G	U	A	L	

		B			
		R	O	S	
		O		A	
I	G	U	A	L	

SEGONA PART

En aquesta part continuarem dissenyant una estructura de dades per emmagatzemar un conjunt de campionats del popular joc ”Apalabrados”. Ara ampliarem les funcionalitats i treballarem amb volums d’informació grans que requeriran la utilització d’estructures de dades dels mòduls 4 al 7. Concretament, per a la realització de l’exercici considereu:

De la PRIMERA PART:

- El nombre de campionats C que guardarem a la prova pilot és variable i relativament petit, d’uns centenars.
- El nombre de lletres diferents disponibles L serà de 29, cada lletra té una puntuació associada en funció de lo comú que sigui (per exemple, la ‘A’ donarà 1 punt, la ‘H’ donarà 8 punts, etc).
- El nombre de jugadors d’una partida JP serà de 2.
- El nombre de fitxes disponibles F inicialment en una partida serà fix, 100. No confongueu les lletres disponibles a un campionat amb les fitxes d’una partida, les fitxes de les partides són la suma de totes les ocurrències de lletres disponibles per jugar. Tingueu en compte també que hi ha una proporció més elevada de fitxes per les lletres més comuns (per exemple, 12 fitxes de la lletra ‘E’, 2 fitxes de la lletra ‘B’, etc...). Podeu trobar la relació de fitxes i lletres en el següent enllaç: http://ca.wikipedia.org/wiki/Scrabble
- Cada jugador tindrà un màxim de 7 lletres per fer una jugada.
- El nombre de jugades GP d’una partida és variable i relativament petit.

Canvis respecte la PRIMERA PART:

- El nombre de partides P d’un campionat també és va riable i molt gran.
- El nombre de paraules PA que formaran part del diccionari de paraules vàlides d’un campionat serà molt gran i de mida coneguda.
- El tauler de cada partida tindrà 15 files i 15 columnes. Considereu que sí hi ha caselles especials (lletra per 2 o per 3, paraula per 2 o per 3).
- Si un jugador posa les set fitxes en una mateixa jugada, obté 40 punts extra.

Nou a la SEGONA PART:

- El nombre de jugadors J donats d’alta serà molt gran i en constant augment.
- El nombre de jugadors inscrits I a un campionat serà variable i, en alguns campionats, pot ser molt gran.

Exercici 5

Especifiqueu un TAD Apalabrados per guardar els campionats que permeti:

De la PRIMERA PART:

- Crear un campionat nou, el campionat tindrà un nom i aquest serà únic.
- Afegir una paraula al diccionari de paraules que es farà servir a tot el campionat.
- Afegir una lletra amb la seva puntuació i el nombre de fitxes d'aquesta lletra que hi haurà disponibles. Aquesta associació es farà servir a tot el campionat.
- Afegir una fitxa a la partida d'un campionat. No es poden afegir fitxes a una partida començada ni més fitxes d'una lletra que les màximes indicades (a la mateixa lletra).
- Començar la partida. Inicialitza una partida d'un campionat i assigna a cada jugador les seves fitxes inicials. El repartiment de fitxes consisteix en agafar una fitxa associada a la partida (l'última inserida) per cada jugador de manera alternativa fins completar les seves 7 fitxes.
- Esbrinar quantes vegades s'ha utilitzat una determinada paraula en el campionat.
- Esbrinar quina paraula ha obtingut la puntuació més alta al campionat.

Canvis respecte la PRIMERA PART:

- Crear una nova partida, la partida tindrà un nom que la identificarà i, en el moment de crear-la, s'especificarà el NIF dels dos jugadors que la jugaran i el campionat en el que participen. Cal que els jugadors estiguin inscrits al campionat.
- Fer una jugada a una partida d'un campionat. La partida no pot estar finalitzada. Caldrà especificar el jugador que fa la jugada, la fila i columna on comença la paraula, l'orientació (horitzontal –d'esquerra a dreta- o vertical –de dalt a baix-) i les lletres necessàries per fer la paraula. Cal comprovar que la paraula sigui vàlida, que el jugador té totes les fitxes necessàries per fer-la, calcular la puntuació de la jugada i actualitzar les lletres disponibles del jugador i la seva puntuació. Per actualitzar la puntuació cal tenir en compte els punts de les paraules generades perpendicularment a la nova paraula afegida. NOTA: veure els aclariments al final d'aquest document.

Nou a la SEGONA PART:

- Donar d'alta un jugador al sistema a partir del seu NIF i nom. En cas d'existir un altre jugador amb el mateix NIF, en modificarem les seves dades.
- Inscriure un jugador a un campionat a patir del seu NIF. Cal que el jugador estigui donat d'alta al sistema. Si el jugador ja estava inscrit, eliminarem la seva inscripció. Cal que les inscripcions del campionat estiguin obertes.
- Tancar la inscripció d'un campionat.

- Consultar les fitxes d'un jugador, en el moment actual d'una partida d'un campionat.
- Canviar una fitxa d'un jugador d'una partida d'un campionat.
- Finalitzar una partida d'un campionat. Cal que un dels dos jugadors s'hagi quedat sense fitxes.
- Consultar els punts dels dos jugadors d'una partida d'un campionat. Aquesta consulta es pot fer en qualsevol moment de la partida, finalitzada o no.
- Consultar la classificació d'un campionat ordenant els jugadors inscrits pel nombre de partides guanyades i els punts obtinguts (per desempatar jugadors que hagin guanyat el mateix nombre de partides). No es tindran en compte els punts de les partides no finalitzades. En cas d'empat no importa l'ordre. Cal que aquesta operació sigui especialment ràpida.

Apartat a)

Dóna la signatura del TAD Apalabrados dels mètodes nous de la SEGONA PART. És a dir, indica el nom que donaries a les operacions encarregades de cada funcionalitat requerida. Indica, també, quins caldria que fossin els paràmetres d'entrada i quina la sortida en cas que es necessités.

Apartat b)

Fes l'especificació contractual de les operacions del TAD Apalabrados, modificades o noves a la PAC2. En la redacció de l'especificació pots fer servir, si et cal, qualsevol de les operacions del TAD. Pren com a model l'especificació de l'apartat 1.2.3 del Mòdul 1 dels materials docents. Es valorarà especialment la concisió (absència d'elements redundants o innecessaris), precisió (definició correcta del resultat de les operacions), completesa (consideració de tots els casos possibles en què es pot executar cada operació) i manca d'ambigüitats (coneixement exacte de com es comporta cada operació en tots els casos possibles) de la solució. És important respondre aquest apartat usant una descripció condicional i no procedimental. L'experiència ens demostra que no sempre resulta fàcil distingir entre les dues descripcions, és per això que fem especial èmfasi insistint que poseu molta atenció a les vostres definicions.

A títol d'exemple indicarem que la descripció condicional (la correcta a utilitzar en el contracte) d'omplir un got buit amb aigua seria:

> @pre el got es troba buit.
>
> @post el got és ple d'aigua.

En canvi una descripció procedimental (incorrecta per utilitzar en el contracte) tindria una forma semblant a:

> @pre el got hauria de trobar-se buit, si no es trobés buit s'hauria de buidar.
>
> @post s'acosta el got a l'aixeta i s'hi tira aigua fins que estigui ple.

Cal també tenir en compte que un contracte hauria de disposar d'invariant sempre que aquesta fos necessària per descriure el TAD.

Exercici 6

A l'Exercici 5 heu definit l'especificació del TAD Apalabrados. Ara us demanem que feu el disseny de les estructures de dades que formaran aquest TAD. Dissenyeu, doncs, el sistema per tal que sigui el màxim d'eficient possible, tant a nivell d'eficiència espaial com temporal, tenint en compte els volums d'informació i les restriccions especificades a l'enunciat.

Tingueu en compte només les operacions que es demanen a l'enunciat a l'hora de fer aquest disseny.

Apartat a)

Dubtem entre utilitzar una llista encadenada, una taula de dispersió o un AVL per emmagatzemar els jugadors inscrits a un campionat. Justifiqueu quina creieu que és la millor opció.

Apartat b)

Dubtem entre utilitzar un vector, una llista encadenada o un AVL per emmagatzemar les partides que ha fet un jugador. Justifiqueu quina creieu que és la millor opció.

Apartat c)

Dubtem entre utilitzar un vector ordenat, una llista encadenada ordenada o una taula de dispersió per emmagatzemar les paraules del diccionari d'un campionat. Justifiqueu quina creieu que és la millor opció.

Apartat d)

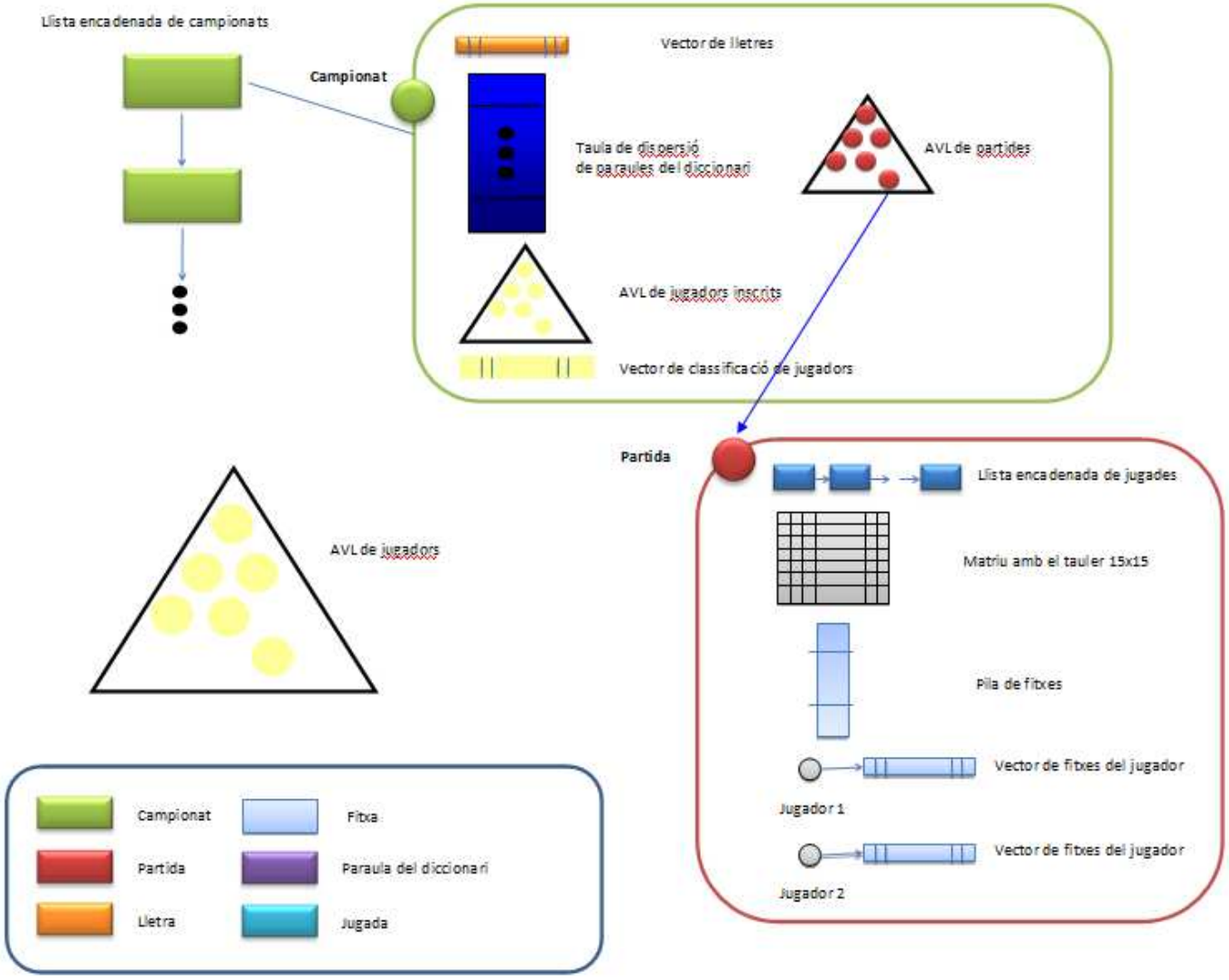
Dubtem entre utilitzar una cua, una taula de dispersió o un AVL per emmagatzemar les partides d'un campionat. Justifiqueu quina creieu que és la millor opció.

Apartat e)

Dubtem entre utilitzar un vector ordenat, una llista encadenada ordenada o un AVL per guardar la classificació d'un campionat. Justifiqueu quina creieu que és la millor opció.

Apartat f)

Feu un dibuix de l'estructura de dades global pel TAD Apalabrados on es veginclarament les estructures de dades que trieu per representar cada una de les parts i les relacions entre elles. Cal que feu el dibuix de l'estructura complerta, amb totes els estructures que us permetin implementar les operacions definides a l'especificació.


Llista encadenada de campionats
Campionat
Vector de lletres
Taula de dispersió de paraules del diccionari
AVL de partides
AVL de jugadors inscrits
Vector de classificació de jugadors
Partida
Llista encadenada de jugades
Matriu amb el tauler 15x15
Pila de fitxes
Vector de fitxes del jugador
Jugador 1
Vector de fitxes del jugador
Jugador 2
AVL de jugadors
Campionat
Fitxa
Partida
Paraula del diccionari
Lletra
Jugada

Exercici 7

A l'Exercici 5 heu definit l'especificació del TAD Apalabrados amb les seves operacions i a l'Exercici 6 heu triat les estructures de dades per cada part del TAD. En aquest exercici us demanem que us fixeu en els algorismes que us serviran per implementar algunes de les operacions especificades i en l'estudi d'eficiència de les mateixes. Tingueu en compte que la implementació de les operacions va estretament lligat a l'elecció de les estructures de dades que hagueu fet.

Apartat a)

Descriviu i feu l'estudi d'eficiència de l'operació que hagueu definit per fer una jugada a una partida. Per fer-ho heu de descriure breument el seu comportament indicant els passos que la componen (amb frases com ara: "inserir en l'arbre AVL / esborrar de la taula de dispersió / consulta del piló / ordenar el vector..."), dient l'eficiència asimptòtica de cada pas i donant l'eficiència total de l'operació. No heu de fer el pseudocodi, només descriure els passos.

Apartat b)

Descriviu i feu l'estudi d'eficiència de l'operació que hagueu definit per finalitzar una partida i obtenir la classificació d'un campionat. Igual que a l'exercici anterior, heu de descriure breument el seu comportament indicant els passos que la componen (amb frases com ara: "inserir en l'arbre AVL / esborrar de la taula de dispersió / consulta del piló / ordenar el vector..."), dient l'eficiència asimptòtica de cada pas i donant l'eficiència total de l'operació. No heu de fer el pseudocodi, només descriure els passos.

Apartat c)

Descriviu i feu l'estudi d'eficiència de l'operació que hagueu definit per canviar una fitxa d'un jugador. Heu de descriure breument el seu comportament indicant els passos que la componen (amb frases com ara: "inserir en l'arbre AVL / esborrar de la taula de dispersió / consulta del piló / ordenar el vector..."), dient l'eficiència asimptòtica de cada pas i donant l'eficiència total de l'operació. No heu de fer el pseudocodi, només descriure els passos.

Exercici 8

Indica quins dels TADs de la biblioteca de TADs de l'assignatura et semblen més adients per utilitzar-los en la implementació de cada una de les estructures de dades definides pel TAD Apalabrados.

Aclariment: Suposant que el tauler es troba en la situació indicada a l'esquerra, el jugador pot jugar "BROU" des de la posició 2,3 en vertical, però només necessita les lletres "B" i "O", a l'aprofitar les lletres "R" i "U" que ja es troben al tauler. És a dir, cal comprovar que "BROU" (la paraula que juga el jugador) és una paraula vàlida i que el jugador té les lletres "B" i "O" (les que necessita per completar la paraula que vol jugar). En aquest cas, el jugador només fa una paraula.

		R	O	S	
				A	
I	G	U	A	L	

		B			
		R	O	S	
		O		A	
I	G	U	A	L	

Igualment, si ara l'altre jugador juga la paraula "MAR" a la posició 6,4 en horitzontal, de fet està generant tres paraules noves, la ja esmentada "MAR" però també "AM" i "SALA", resultat d'afegir la lletra "M" i la lletra "A" a "A" i "SAL" respectivament. Observeu que potser "SAL" era una paraula jugada per algun jugador però, en canvi, la paraula "A" mai havia estat jugada, sinó que era un tros d'una altra paraula (en aquest cas, "IGUAL").

		B			
		R	O	S	
		O		A	
I	G	U	A	L	

		B			
		R	O	S	
		O		A	
I	G	U	A	L	
			M	A	R

Finalment, recordeu que ara cal tenir en compte si una o més fitxes cauen en una casella que multiplica el seu valor i/o el de les paraules resultants.

PRIMERA PART

Exercici 1

apartat a)

void crearCampionat(String idCampionat)

void afegirParaula(String idCampionat, String paraula)

void afegirLletra(String idCampionat, String lletra, int punts, int numFitxes)

void crearPartida(String idCampionat, String nom, String primerJugador, String segonJugador)

void afegirFixa(String idCampionat, String nomPartida, String lletra)

void comencarPartida(String idCampionat, String nomPartida)

void ferJugada(String idCampionat, String nomPartida, int jugador, int fila, int columna, int orientacio, String paraula)

int numeroVegades(String idCampionat, String paraula)

String paraulaAmbPuntuacioMesAlta(String idCampionat)

apartat b)

@pre no existeix cap campionat amb l'identificador especificat

@post existeix un campionat buit amb l'identificador especificat

void crearCampionat(String idCampionat)

@pre existeix el campionat i el seu diccionari de paraules no conté la paraula especificada

@post el diccionari de paraules del campionat conté la paraula especificada

void afegirParaula(String idCampionat, String paraula)

@pre existeix el campionat, la seva llista de lletres no conté la lletra especificada i la suma total de fitxes no supera les 100

@post la llista de lletres conté la lletra especificada amb la puntuació i el nombre de fitxes. La nova fitxa s'afegeix al darrera

void afegirLletra(String idCampionat, String lletra, int punts, int numFitxes)

@pre existeix el campionat, no existeix cap partida amb el nom especificat

@post existeix una partida no començada entre el primer jugador i el segon jugador

void crearPartida(String idCampionat, String nom, String primerJugador, String segonJugador)

@pre existeix la partida, no ha començat i el nombre de fitxes afegides per la lletra especificada es

inferior al màxim permès

@post s'ha incrementat en una unitat el nombre de fitxes disponibles en la partida per la lletra especificada

void afegirFixa(String idCampionat, String nomPartida, String lletra)

@pre existeix el campionat i la partida i aquesta no esta començada

@post la partida està començada i cada jugador té les set fitxes inicials

void comencarPartida(String idCampionat, String nomPartida)

@pre existeix el campionat i la partida i aquesta esta començada.

@post s'ha enregistrat la jugada del jugador al tauler de la partida, s'han tret les fitxes de les lletres jugades al jugador, se li ha actualitzat la puntuació i se li han reposat les fitxes

void ferJugada(String idCampionat, String nomPartida, int jugador, int fila, int columna, int orientacio, String paraula)

Nota: També seria correcte especificar com a precondició les condicions que fan que una jugada sigui correcta. En el nostre cas hem considerat que aquestes condicions es comproven a la operació , per tant, no apareixen a la precondició.

@pre existeix el campionat

@post el valor retornat és el nombre de cops que la paraula ha sortit durant el campionat

int numeroVegades(String idCampionat, String paraula)

@pre existeix el campionat

@post el valor retornat la paraula amb puntuació més alta al campionat

String paraulaAmbPuntuacioMesAlta(String idCampionat)

Nota: A tot l'exercici hem especificat com a precondicions algunes comprovacions que es podrien fer dins les operacions. També seria correcte considerar que les comprovacions les fa la mateixa operació, en aquest cas, les precondicions no haurien d'aparèixer i les potscondicions haurien d'ampliarse (o afeblir-se) per definir l'estat quan les condicions fallen.

Exercici 2

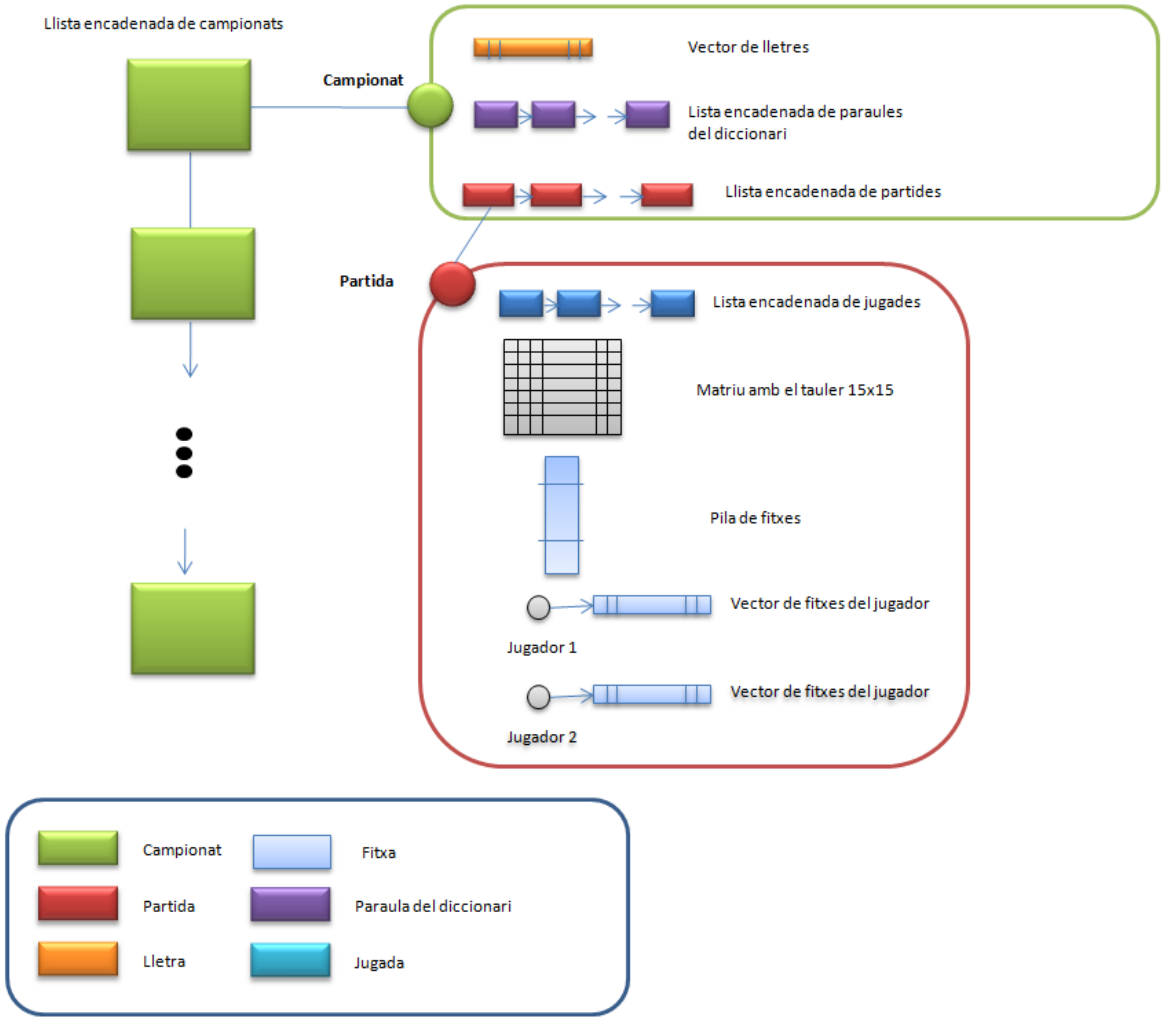
Apartat a)

L'enunciat ens diu que el nombre de jugades GP d'una partida variable i relativament petit, per volums de dades petits i variables, la millor opció és fer servir una estratègia basada en memòria dinàmica per no malbaratar espai. Com que a l'enunciat no ens demanen cap operació que requereixi la llista de jugades ordenada triarem una llista encadenada per guardar les jugades d'una partida.

Apartat b)

L'enunciat ens diu que el nombre de fitxes F disponibles per una partida es fixes i petit: 100 fitxes. Per volums de dades petits i fixes, la millor opció és fer servir una estratègia basada en vector per no malbaratar espai pels apuntadors. Pel repartiment inicial de fitxes l'enunciat ens diu que fem servir una estratègia basada en una pila i no especifica l'estratègia que es farà servir per anar repartint les fitxes als jugadors després de cada jugada així que podem triar una pila implementada amb un vector per guardar les fitxes d'una partida.

Apartat c)



Apartat d)

- Per guardar els campionats triem una llista encadenada ja que el número de campionats és variable i relativament petit, d'uns centenars. Tot i que necessitem accés directe el nombre d'elements es prou petit com per admetre cerques lineals.
- Per les partides també triem una llista encadenada, el raonament és el mateix que pels campionats.
- Per les paraules del diccionari també triem una llista encadenada, el raonament és el mateix que pels campionats i les partides.
- Per les lletres triem un vector ja que el número es petit però no es variable. També necessitarem accés directe però amb 29 elements en podem permetre les cerques lineals.
- Per les fitxes d'una partida triem una pila implementada amb un vector ja que cada partida tindrà 100 fitxes i necessitem una estructura basada en una pila per repartir les fitxes inicials i les fitxes de cada jugada. En aquest cas no necessitem accés directe.

- Pels dos jugadors d'una partida no necessiten cap estructura addicional ja que l'enunciat no especifica cap operació que involucri a tots els jugadors d'un campionat, es guardaran doncs com a atributs de cada partida.
- Les lletres que té cada jugador en un moment determinat se de 7, farem servir doncs un vector amb les lletres disponibles de cada jugador. Igual que abans, encara que es requereix accés directe, amb 7 elements ens podem permetre cerques lineals.
- Per guardar el tauler triarem una matriu de 15x15.
- *Per guardar les jugades d'una partida triem una llista encadenada ja que el número de jugades és variable i relativament petit. No necessitem accés directe però si un recorregut per comptar el nombre de vegades que apareix una paraula.*

Exercici 3

Apartat a)

Simplificant-ho molt, l'operació *ferJugada* hauria de:

- Comprovar que la paraula existeixi al diccionari => O(PA)
- Recuperar el campionat a la llista de campionats => O(C)
- Recuperar la partida de la llista de partides del campionat => O(P)
- Comprovar quines de les lletres de la paraula ja estan al tauler => O(1)
- Comprovar que el jugador té totes les fitxes amb les lletres que falten per completar la paraula => O(1)
- Posar les fitxes al tauler => O(1)
- Comptar els punts de la paraula, consultant la puntuació de cada fitxa posada al vector de lletres => O(1)
- Actualitzar, si cal, la paraula amb puntuació més alta del campionat => O(1)
- Afegir al vector de fitxes del jugador tantes fitxes com hagi posat al tauler. Les fitxes s'agafen del la pila de fitxes => O(1)

Per tant, el cost total de la operació seria de O(PA+C+P)

Apartat b)

Si no contemplem un atribut que guardi la paraula amb la puntuació més alta a l'hora de fer una jugada l'operació seria força costosa ja que hauria d'iterar per totes les jugades de totes les partides de tots els campionats calculant la puntuació de les paraules de cada jugada fins a trobar la que hagués proporcionar una puntuació més elevada.

Per la prova pilot que us em plantejat a la PAC1 encara podria ser admissible però quan el volum de dades cresqués una mica us caldria replantejar la operació ja que seria molt ineficient.

Sembla molt més adient guardar a nivell de campionat un atribut amb la paraula amb més puntuació i anar actualitzant aquest atribut a la operació *ferJugada*, amb aquesta estratègia el cost de la operació d'esbrinar la paraula amb puntuació més elevada és redueix a consultar aquest atribut O(1).

Apartat c)

Per tal de reproduir les partides caldria guardar les jugades de forma ordenada, per tant, una possible solució seria canviar la llista encadenada de jugades per una llista encadenada ordenada segons el número de jugada. Si no teníeu el numero de jugada especificat a la operació *ferJugada* també us caldria afegir-lo (o guardar un comptador intern) per tal que les jugades es puguin ordenar segons aquest criteri.

Exercici 4

- Per guardar els campionats triem una llista encadenada, el TAD més adient seria una *LlistaEncadenada*.
- Per les partides també triem una *LlistaEncadenada*.
- Per les paraules del diccionari també triem una *LlistaEncadenada*.
- Per les lletres triem la classe un array de Java ja que a la biblioteca de TADs no hi ha cap implementació de contenidor seqüencial afitat.
- Per les fitxes d'una partida triem una *PilaVectorImpl*.
- Pels dos jugadors d'una partida no necessiten cap estructura addicional.
- Les lletres que té cada jugador triem una array de Java.
- Per guardar el tauler triarem un array de java bidimensional.
- *Per guardar les jugades d'una partida triem una LlistaEncadenada..*

SEGONA PART

Exercici 5

Apartat a)

void addPlayer(String NIF, String nom);
void addOrRemoveInscription(String idChampionship, String NIF);
void closeInscriptions(String isChampionship);
Iterator<Card> cards(String idChampionship, String gameId, int playerNumber);
void changeCard(String idChampionship, String gameId, int playerNumber, Card card);
void finishGame(String idChampionship, String gameId);
Points[] getPoints(String idChampionship, String gameId);
Iterator<Player> classification(String idChampionship);

Apartat b)

@pre cert.
@post els jugadors registrats al sistema seran el mateixos d'abans de l'operació més el nou jugador, o el jugador amb nif NIF tindrà les dades canviades.
void addPlayer(String NIF, String nom);

@pre el campionat existeix, la preinscripció està oberta, i hi ha un jugador amb NIF registrat al sistema.
@post si el jugador amb nif NIF no estava inscrit al campionat, ara ho estarà. Pel contrari, si ja estava inscrit ara no ho estarà.
void addOrRemoveInscription(String idChampionship, String NIF);

@pre el campionat existeix i la inscripció està oberta.
@post el campionat té la inscripció tancada.
void closeInscriptions(String isChampionship);

@pre el campionat i el joc existeixen, i playerNumber val 0 o 1.
@post retorna les fitxes del jugador de la partida del campionat indicat.
Iterator<Card> cards(String idChampionship, String gameId, int playerNumber);

@pre el campionat i el joc existeixen, playerNumber val 0 o 1 i card és una de les fitxes del jugador.
@post El jugador tindrà les mateixes fitxes excepte card i, en lloc seu, tindrà una de les fitxes

pendents d'adjudicar del joc X. Les fitxes pendents d'adjudicar del joc seran les mateixes exceptuant X que haurà canviat per card.
void changeCard(String idChampionship, String gameId, int playerNumber, Card card);

@pre el campionat i el joc existeixen i no queden fitxes pendent d'adjudicar i, o bé el jugador1 o el jugador2, no tenen fitxes.
@post el joc està finalitzat.
void finishGame(String idChampionship, String gameId);

@pre el campionat i el joc existeixen.
@post retorna els punts dels dos jugadors.
Points[] getPoints(String idChampionship, String gameId);

@pre el campionat existeix
@post retorna un iterador per recórrer els jugadors del campionat ordenats pels punts obtinguts. En cas d'empat no importa l'ordre.
Iterator<Player> classification(String idChampionship);

Exercici 6

Apartat a)

El nombre de jugadors inscrits a un campionat és molt gran i variable. Així doncs necessitem una estructura no afitada i, per tant, descartem les taules de dispersió. Una llista encadenada seria una opció vàlida, però comportaria costos lineals cada cop que necessitéssim verificar que un jugador està inscrit a un campionat (bàsicament a addGame i addOrRemoveInscription). Per això ens decantem per un AVL.

Apartat b)

Aquesta informació no és necessària per poder implementar cap dels mètodes especificats a l'enunciat. Així doncs, no guardarem aquesta informació.

Apartat c)

El nombre de paraules serà gran i conegut. Podem utilitzar qualsevol de les estructures proposades, però una llista encadenada ordenada ens portaria a costos lineals cada cop que necessitéssim fer una consulta per comprovar l'existència d'una paraula. Un vector ordenat ens portaria a costos logarítmics i una taula de dispersió a costos constants. Optem per la taula de dispersió.

Apartat d)

Una cua només ens permetria accedir a la primera partida afegida, i molts mètodes necessiten accedir a les partides a partir del seu id. El nombre de partides és gran i variable i això ens impedeix d'utilitzar estructures afitades com les taules de dispersió. Així, la única opció vàlida serà l'AVL.

Apartat e)

El nombre de jugadors d'un campionat és gran i variable, però un cop començat el campionat la inscripció es tanca i, per tant, el nombre de jugadors passa a ser gran i constant. Per tant, pel volum de dades a guardar, totes les opcions són vàlides.

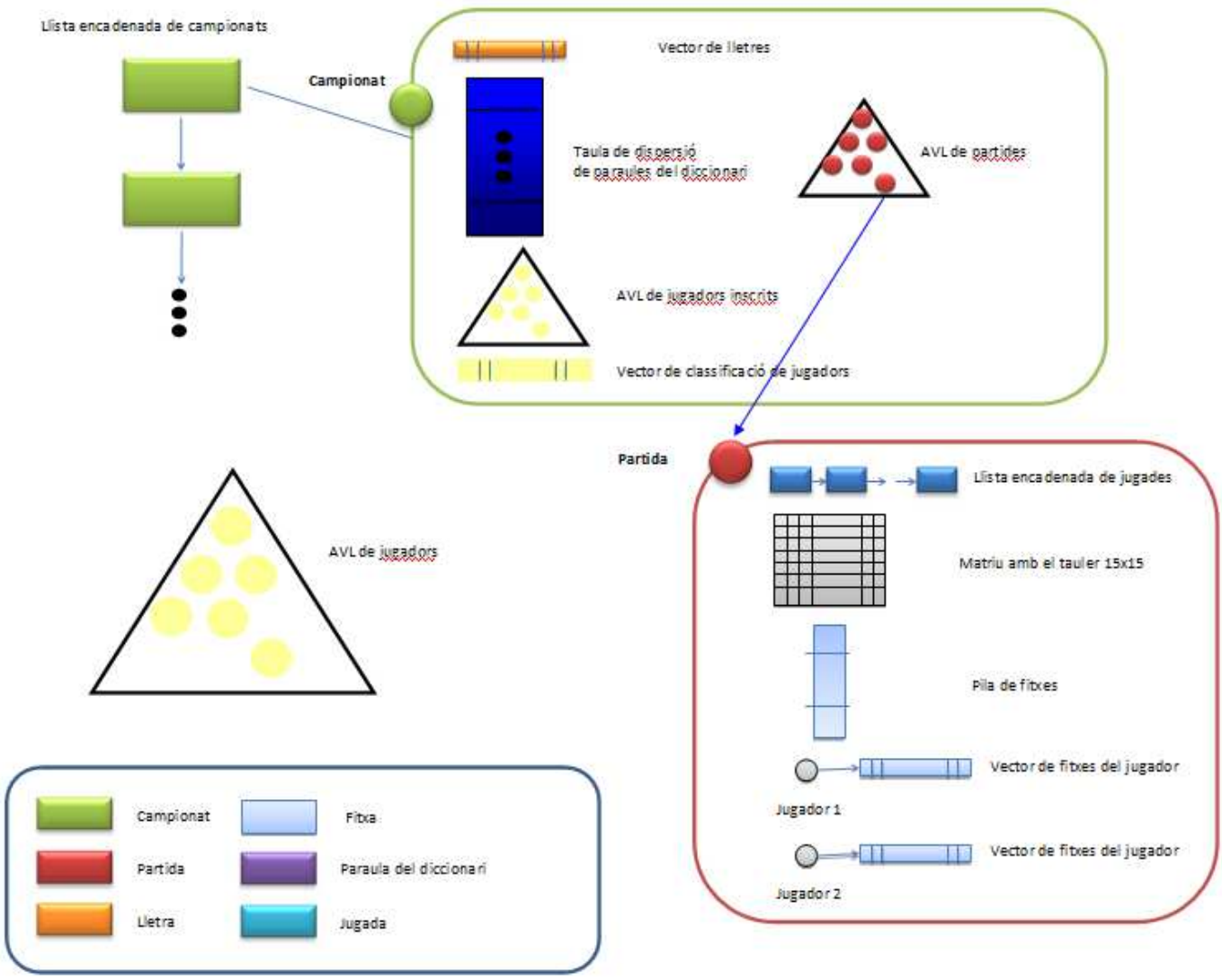
Per poder obtenir un iterador de la classificació cal que tinguem els jugadors ordenats pels seus punts. Això no és possible utilitzant els AVL que tenim implementats a la biblioteca, ja que no permeten claus repetides.

Per evitar aquesta limitació, podem definir la clau com els punts + el NIF. D'aquesta manera mantenim l'ordre i evitem les claus repetides.

Les llistes encadenades ordenades ens portarien a costos lineals per cercar un jugador, mentre que un vector ordenat o l'AVL seria logarítmic.

Per actualitzar la posició d'un jugador en un AVL cal eliminar-lo, sumar els punts, i tornar-lo a afegir => O(log I). Si utilitzem un vector ordenat, en el pitjor dels casos haurem de moure tots els jugadors 1 cel·la (quan el darrer jugador de la classificació guanyi tants de punts que passi a ser el primer). Però aquesta situació serà poc habitual, i només podrà passar en l'inici de la partida, quan tots els jugadors tinguin 0 punts. De fet, normalment els moviments que faran els jugadors dins de la classificació seran petits i, per tant, el cost d'utilitzar vectors ordenats serà inferior als AVL.

Apartat f)



Exercici 7

Apartat a)

- Comprovar que la paraula existeixi al diccionari => O(1)
- Recuperar el campionat a la llista de campionats => O(C)
- Recuperar la partida de la llista de partides del campionat => O(log P)
- Comprovar quines de les lletres de la paraula ja estan al tauler => O(1)
- Comprovar que el jugador té totes les fitxes amb les lletres que falten per completar la paraula => O(1)
- Posar les fitxes al tauler => O(1)
- Comptar els punts de la paraula, consultant la puntuació de cada fitxa posada al vector de lletres => O(1)
- Per cada fitxa posada, comprovar si es forma una paraula perpendicularment.
  - Cercar la paraula perpendicular al tauler =>O(1)
  - Comprovar si la paraula existeix al diccionari => O(1)
  - Comptar els punts =>O(1)
  - Total: O(1) * pel número de fitxes posades => O(1) (considerem el nombre de fitxes posades un número prou petit comparat amb les altres).
- Actualitzar, si cal, la paraula amb puntuació més alta del campionat => O(1)
- Afegir al vector de fitxes del jugador tantes fitxes com hagi posat al tauler. Les fitxes s'agafen del la pila de fitxes => O(1)

Per tant, el cost total de l'operació seria de O(C+log P)

Apartat b)

L'enunciat ens demana que l'operació per obtenir la classificació sigui el més eficient possible. Per aconseguir cost O(1) cal que sempre tinguem la classificació ja ordenada i que la feina la fem en els mètodes que modifiquen la classificació.

L'únic mètode que altera la classificació és la finalització d'una partida. Per tant, serà en aquest mètode que sumarem els punts al jugador:

void finishGame(String idChampionship, String gameId);

- Cercar campionat => O(C)
- Cercar partida =>O(log P)
- Per cada jugador:
  - Accedir a les dades del jugador => O(1) (si tenim una referència a l'objecte jugador des de partida)

- Consultar els seus punts (atribut de Jugador) => O(1)
- Localitzar el jugador a la classificació (clau: punts+NIF) =>O(log I)
- Actualitzar els punts al jugador => O(1)
- Moure el jugador a la classificació =>O(1) (veure explicació de l'exercici 2e)

Per tant, el cost total de l'operació seria de O(C+log P+log I)

Apartat c)

void changeCard(String idChampionship, String gameId, int playerNumber, Card card);

- Cercar campionat => O(C)
- Cercar partida =>O(log P)
- Cercar card al vector de fitxes del jugador => O(7) = O(1)
- Sortejar una fitxa de la pila de fitxes =>O(1) (comporta afegir un nou mètode a la Pila!)
- Intercanviar card per la fitxa sortejada =>O(1)

Per tant, el cost total de l'operació seria de O(C+log P+log I)

Exercici 8

- Per les partides i les inscripcions triem una DiccionariAVLImpl
- Per les paraules del diccionari triem una TaulaDispersio.
- Per les lletres triem la classe un array de Java ja que a la biblioteca de TADs no hi ha cap implementació de contenidor seqüencial afitat.
- Per la classificació un DiccionariVectorImpl.
- Per les fitxes d'una partida triem una PilaVectorImpl ampliada per poder canviar fitxes.
- Pels dos jugadors d'una partida no necessiten cap estructura addicional.
- Les lletres que té cada jugador triem una array de Java.
- Per guardar el tauler triarem un array de java bidimensional.
- Per guardar els campionats triem una llista encadenada, el TAD més adient seria una LlistaEncadenada.
- Per guardar les jugades d'una partida triem una LlistaEncadenada.