Tema 1 - Exercici Recuperació 3

Volem implementar una versió simplificada del joc de la Oca. En aquesta versió simplificada el joc tindrà menys caselles que en la versió habitual i les caselles seran només de 5 tipus:

- Casella normal: no té cap acció associada.
- Oca: el jugador salta fins a la següent casella on hi hagi una oca i torna a tirar. Si no hi ha cap més oca des d'aquella posició fins a l'úlitma casella, es queda a la mateixa casella i no torna a tirar
- Pou: el jugador que hi cau es queda dos torns sense tirar.
- Mort: el jugador ha de tornar a la casella inicial.
- Casella final: última casella del tauler. El jugador que hi arriba primer guanya la partida. S'hi ha d'arribar amb un valor exacte del dau. Si la suma del valor del dau a la posició actual del jugador dóna un valor més gran que la posició de la casella final, el jugador es queda a la seva posició actual, sense poder-se moure.

Per implementar el joc voldrem crear tres classes: `Jugador`, `Casella` i `Tauler`.

Tema 1 - Exercici Recuperació 3

La **classe `Jugador`** servirà per guardar la informació necessària de cadascun dels jugadors que participen a la partida. Haurà de poder guardar a quina posició del tauler està el jugador (valor entre 1 i nº màxim de caselles), si pot tirar o no al següent torn i, en cas de que no pugui tirar, quants torns ha d'estar sense tirar, i si és el guanyador de la partida. A nivell d'interfície pública haurà de tenir, com a mínim, els mètodes que facin falta per inicialitzar/recuperar/modificar aquests valors.

La **classe `Casella`** guarda la informació bàsica de cada casella del joc. Haurà de poder guardar la posició i el tipus de la casella. A nivell d'interfície pública haurà de tenir, com a mínim, els mètodes que facin falta per inicialitzar/recuperar/modificar aquests valors.

A més a més, en alguna de les dues classes hi haurem d'afegir la funcionalitat de gestionar les accions que s'han de fer quan el jugador arriba a una casella, en funció del tipus de la casella.

Tema 1 - Exercici Recuperació 3

La **classe `Tauler`** haurà de tenir els atributs necessaris per poder guardar totes les caselles del joc (fins a un màxim de 63), les dades de tots els jugadors que hi participen (fins a un màxim de 4) i quin és el jugador que té el torn en cada moment del joc. Haurà de tenir com a mètodes:

- Un constructor per defecte que inicialitzi un tauler sense caselles ni jugadors.
- Un mètode inicialitza amb la capçalera següent:

```
void inicialitza(int tipusCaselles[], int nCaselles, int nJugadors);
```

Aquest mètode ha d'inicialitzar totes les caselles del tauler a partir del nº de caselles i l'array `tipusCaselles` que rep com a paràmetres. L'array conté valors enters indicant el tipus de totes les caselles des de la casella 1 fins a l'última. També ha de fixar i inicialitzar el nº de jugadors al valor que es passa com a paràmetre, i donar el torn actual al primer jugador.

Els valors enters que indiquen el tipus de cada casella queden fixats per aquestes constants que estan declarades al fitxer `casella.h`:

```
const int NORMAL = 1;
const int OCA = 2;
const int POU = 3;
const int MORT = 4;
const int FINAL = 5;
```

Tema 1 - Exercici Recuperació 3

- Un mètode per simular un torn del joc amb la capçalera següent:

```
void tornJoc(int valorDau);
```

Aquest mètode ha de servir per actualitzar l'estat de la partida en funció del jugador que tingui el torn. Si el jugador que té el torn no pot tirar, no es fa res, però es disminueix el nº de torns sense tirar del jugador.

Si el jugador actual sí que pot tirar, el paràmetre `valorDau` tindrà el valor del dau entre 1 i 6. S'ha de calcular la nova posició a la que ha d'anar el jugador i si la posició és més gran que la casella final tampoc es fa res, considerem que el jugador no es pot moure.

Si es pot moure, s'haurà de moure el jugador a la casella corresponent i fer les accions associades al tipus de casella cridant els mètodes de les classes `Casella` i `Jugador` que faci falta. Hem de tenir en compte que si la casella és una oca s'haurà de buscar dins del tauler la següent casella que sigui una oca i moure el jugador a aquesta nova casella, conservant el torn a la següent tirada. Si ja no hi ha més oques abans de la casella final, el jugador no es mou i no conserva el torn.

Al final, si el jugador no conserva el torn s'ha de passar el torn al següent jugador modificant l'atribut corresponent de la classe.

Tema 1 - Exercici Recuperació 3

- Un mètode `getTipusCasella` amb la capçalera següent:

```
int getTipusCasella(int nCasella);
```

Aquest mètode és necessari només pel test del joc i ha de retornar el tipus de la casella de la posició que es passa com a paràmetre (començant per la posició 1).

- Un mètode `getEstatJugador` amb la capçalera següent:

```
int getEstatJugador(int nJugador, int& posicio,
    bool& potTirar, int& nTornsInactiu, bool& guanyador);
```

Aquest mètode és necessari només pel test del joc. Ha de retornar l'estat del jugador (posició del tauler, si pot tirar, nº de torns sense tirar (si no ho pot fer) i si és el guanyador del joc) que es passa com a paràmetre (començant pel jugador 1) als paràmetres per referència `posició`, `potTirar`, `nTornsInactiu` i `guanyador`.

Feu també la **declaració** de les classes **`Casella`** i **`Jugador`** identificant els atributs necessaris per guardar la informació necessària i els mètodes que facin falta per poder donar la funcionalitat necessària per la interfície pública.

Tingueu en compte que els mètodes de les classe `Casella` i `Jugador` han de permetre fer totes les operacions necessàries per inicialitzar el tauler i executar un torn de joc tal com s'ha explicat a la descripció de la classe `Tauler`.