S8-PIG « Programmation des Interfaces graphiques en C++ »

TD #4

1 – Exceptions

Remarque : sur Pédagogie, répertoire « TD2017/Exceptions », vous trouverez les codes des trois exemples présentés en cours.

Partie 1 (exception standard) :

Créer un nouveau projet « damierexc ». Créez une classe « DamierExc » et recopiez la classe « DamierDyn » dans cette classe (n'oubliez pas de changer les noms des méthodes !). Si vous n'êtes pas sûr de votre classe, vous pouvez repartir de Damier3.zip présent sur Pédagogie.

Dans la méthode « Set (l, c, val) », insérez une exception permettant d'interdire l'accès à une case en dehors des dimensions du damier. Ainsi les quelques lignes de code suivantes devront afficher un message d'erreur générique (par exemple « Accès case du Damier incorrecte ») :

```
DamierExc D(3, 5);
D.Set(2, -1, 8);          // → levée d'exception car -1 n'est pas une coordonnée admissible
```

Partie 2 (exception propre) :

On voudrait maintenant limiter la plage de valeurs que l'on peut stocker dans les cases du Damier, par exemple à [0,3]. La Borne (ici « 4 ») sera fournie comme un paramètre supplémentaire au constructeur. Ainsi l'instruction « DamierExc D(3, 5, 4); » permettra de créer un damier de 3 lignes, 5 colonnes et une borne de 4.

Développez une classe exception spécifique appelée « ExceptionDamier », qui sera levée à chaque fois que l'on tente de stocker une valeur interdite :

```
DamierExc D(3, 5, 4);
D.Set(1, 2, 3); // → pas de levée d'exception
D.Init(6);      // → levée d'exception car 6 est différente de 0, 1, 2 et 3
```

Remarque : Si vous additionnez 2 damiers, on peut s'attendre à ce que la nouvelle borne soit la somme des 2 bornes.

La levée d'exception devra afficher :

- La valeur interdite (« ici 6 ») avec la valeur maximale tolérée (ici « 4 »)
- Le nom du fichier et de la méthode qui a levée l'exception (ici fichier « DamierExc.cpp », méthode « Init »). Pour cela, vous pouvez utiliser les macros __FILE__ et __PRETTY_FUNCTION__.

Typiquement :

```
Borne = 4
Valeur rejetée : 10
Fichier : ../DamierException/damierexc.cpp
Fonction : void DamierExc::Set(int, int, int)
```

2 – Héritage

Mettez en œuvre les principes sur l'héritage sur la classe « DamierExc » pour proposer une classe permettant de créer simplement :

- Un damier pour le jeu de dames (10x10 cases) ; le nombre de pièces est limité à deux et donc la borne sera de 3 (pour prendre en compte le fait qu'une case peut ne pas être occupée par l'une des 2 couleurs).
- Un damier pour le jeu de Go (19x19 cases).
- Un damier pour le jeu d'échec (8x8 cases).

Proposez des méthodes d'initialisation du jeu appelée « InitJeu(…) » pour chacun des 3 damiers (cette méthode simulera le placement des pièces au début d'une partie, en remplaçant les pièces par des entiers).

- Comment peut-on obliger toute classe héritant de « DamierExc » à implémenter un méthode « InitJeu(…) » ?
- Peut-on encore créer des objets de type « DamierExc » ?