C++

Projet

Marc Fonvieille et Joël Gay

1 Généralités

L'objectif de ce projet est de mettre en œuvre les connaissances que vous avez acquises durant l'enseignement de la programmation orientée objet appliquée au langage `C++`. Pour ce faire vous aller devoir :

- Comprendre le cahier des charges
- Modéliser ce cahier sous la forme d'un diagramme de classes en réfléchissant à l'implémentation de chaque fonctionnalité
- Commencer enfin l'étape de programmation
- Préparer un `Makefile` pour permettre de compiler votre programme en compilation séparé.

Ce projet **est orienté objet**, c'est-à-dire qu'il vous est demandé de mettre en pratique les notions vues en cours (héritage, encapsulation, classes, polymorphisme, surcharge, etc.).
Le **rapport** est à rendre au plus tard le **vendredi 31 octobre 2025 à 23H59 (Heure de Paris) par mail aux deux enseignants**. Le projet aura une séance de soutenance le **mardi 4 novembre matin** suite à laquelle le code devra être rendu. Tout retard sera pénalisé.

Vous continuerez à travailler sur les exercices de TP **en parallèle** avec le projet. A l'issue de la dernière séance, vous aurez toujours la possibilité de poser des questions par mail. Ce projet doit être réalisé **en binôme** (il peut en **monôme**, mais c'est déconseillé). Aucun groupe de taille plus grande ne sera accepté. Dans le cas d'un binôme, le travail de chacun devra être

apparent dans le rapport mais également dans le code et lors de la soutenance.
Le sujet du projet a été rédigé de manière à décrire les fonctionnalités basiques du système. Nous proposons ensuite toute une série de fonctionnalités supplémentaires. De plus, il n'y a pas de précisions concernant l'implémentation ou la structure interne du projet qui vous sont laissées au choix.

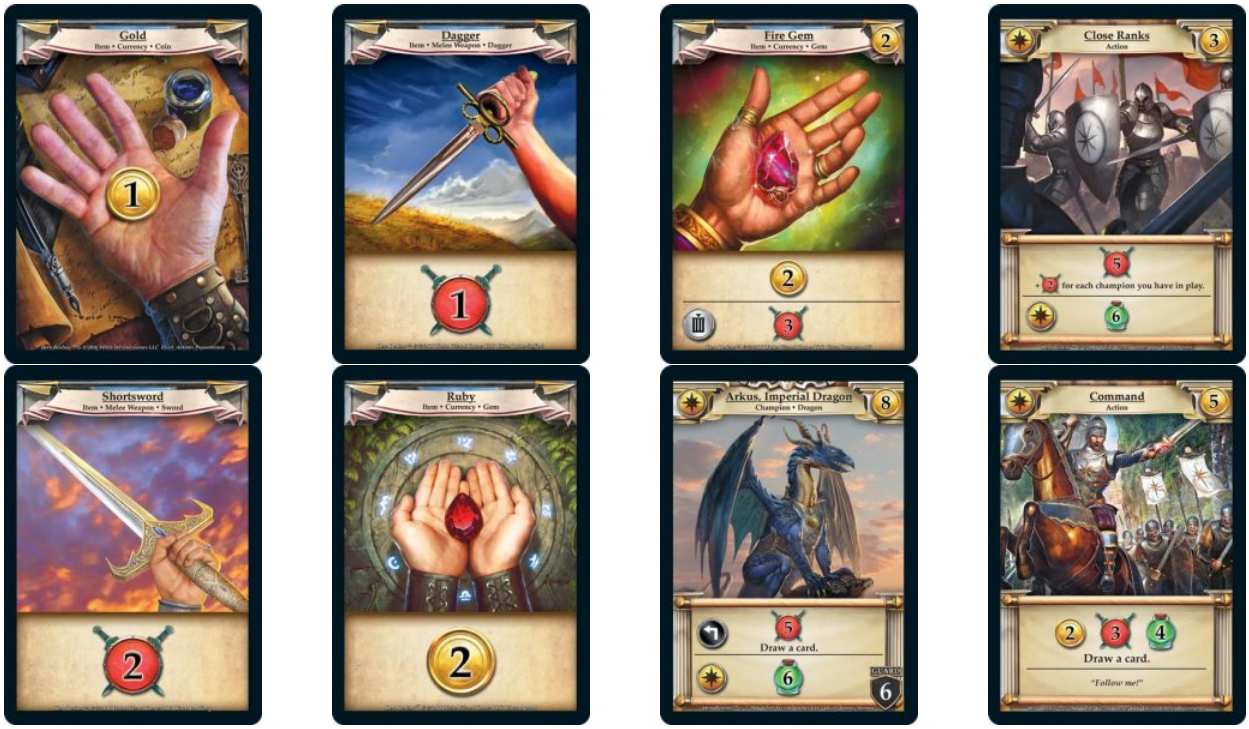
2 Sujet - Le jeu

L'objectif de ce projet est de créer une version du jeu de société de type deckbuilding *Hero Realms*. Il est produit par Iello. Le lien précédent vous permet de récupérer la règle du jeu en pdf. Vous pouvez aussi la récupérer ici.
Vous pouvez visualiser toutes les cartes en allant sur le site officiel et en tapant `base set` dans la barre de recherche. Cela vous donne aussi le nombre de telles cartes qui sont présentes dans la pioche du marché.

Pour tester le jeu en ligne, le site web Tabletopia vous permet de créer un compte (prenez un compte gratuit ! vous n'avez pas besoin de mettre votre carte bancaire !) pour y jouer en ligne. Dans ce site, vous avez une belle IHM, par contre rien n'est automatisé et tout doit être fait à la main.
Vous pouvez aussi télécharger l'application téléphonique. Celle-ci a toutefois le défaut qu'elle ne peut être jouée qu'avec une première extension : celle des decks de classes.

Cette vidéo vous donne une explication en 5 min (les 30 dernières secondes concernent une extension qui n'est pas dans notre jeu de base).

Quelques exemples de cartes




Darian, War Mage
Champion • Human Mage
4
3
or
4
The orcs mocked the man's gibberish at first, but they soon learned to respect Dragontongue.
5


Recruit
Action
2
2
3
+1 for each champion you have in play.
1


Hit Job
Action • Assassin
4
7
Stun target champion.


Influence
Action
2
3
3


Domination
Action
7
6
6
Draw a card.
Prepare a champion.


Bribe
Action
3
3
Put the next action you acquire this turn on top of your deck.


Rake, Master Assassin
Champion • Human Assassin
7
4
You may stun target champion.
His name keeps men of power awake at night.
7


Tyrannor, the Devourer
Champion • Demon
8
4
You may sacrifice up to two cards in your hand and/or discard pile.
Draw a card.
GUARD
6


Kraka, High Priest
Champion • Human Priest
6
2
Draw a card.
Gain 2 for each champion you have in play.
6


Death Threat
Action • Assassin
3
You're Next
1
Draw a card.
Stun target champion.


Smash and Grab
Action • Rogue
6
6
You may put a card from your discard pile on top of your deck.


Elven Curse
Action • Elf Curse
3
6
Target opponent discards a card.
3


Master Weyan
Champion • Human Monk
4
3
+1 for each other champion you have in play.
GUARD
4


Deception
Action
5
2
Draw a card.
Put the next card you acquire this turn into your hand.


Dark Reward
Action
5
3
You may sacrifice a card in your hand or discard pile.
6


Grak, Storm Giant
Champion • Giant
8
6
You may draw a card. If you do, discard a card.
Draw a card, then discard a card.
GUARD
7


Rally the Troops
Action
4
5
5
Prepare a champion.


Fire Bomb
Action
8
8
Stun target champion. Draw a card.
5


Death Touch
Action • Curse
1
2
You may sacrifice a card in your hand or discard pile.
2


Wolf Shaman
Champion • Human Priest
2
2
+1 for each other card you have in play.
4

3 Évaluation et rendus

L'évaluation comportera deux volets : l'évaluation et présentation du code et celle d'un rapport succinct.

3.1 Rapport

Le rapport doit être envoyé pour présenter le projet. Ce **rapport succinct** prendra la forme d'un texte **au format PDF**. Les éléments qui suivent doivent y être présents peu importe le nombre de pages (cela devrait faire autour d'une dizaines de pages sans doute vu les éléments demandés). Il contient notamment :

- Une explication des choix de conception pour répondre au cahier des charges.
- Le diagramme des classes avec commentaires sur le rôle des classes.
- Pour les binômes, une répartition précise du travail
- Un guide d'utilisation du `Makefile` et explication en cas de dépendances ou de packages à installer
- Une explication succincte de chaque fonctionnalité supplémentaire réalisée (suggérée par le sujet ou de votre propre ajout)
- Les difficultés rencontrées, et comment vous les avez surmontés - ou en quoi ils vous ont empêché de terminer une partie du projet.

Pensez à inclure des captures d'écran de votre programme en fonctionnement (en annexe si vous le souhaitez).
En outre, le soin accordé aux schémas, à la présentation, à l'orthographe, sont pris en compte.

3.2 Soutenance

La **soutenance** est une démonstration de code de 10 à 15 minutes (en fonction du nombre de groupes, vos enseignants vous enverrons un timing). Pendant la démonstration, les enseignants testeront votre code, chercheront les limites, poseront des questions de modélisation. Il vous faut donc simultanément pouvoir montrer les fonctionnalités implémentées, mais aussi pouvoir justifier des choix et expliquer le codage de l'application. C'est l'occasion de montrer aussi tout ce que vous avez fait en plus de la base du jeu.

Suite à la soutenance, le code devra être rendu. Tout retard sera pénalisé.
Votre travail (code + rapport) doit être rendu sous forme d'une **archive ZIP**. Vérifiez que le code envoyé inclut seulement les fichiers `.cpp`, `.h(pp)` et `.txt` avec les données nécessaires au fonctionnement de votre programme, ainsi que le `Makefile`. Tout `CMake` sera refusé ! Un programme ne pouvant être compilé chez les enseignants (Linux) ne sera pas accepté.

3.3 Cahier des charges de la base

Le projet doit être indépendant de la plateforme utilisée. Il doit pouvoir être compilé par `g++` sans warning (et évidemment sans erreur). Vous fournirez obligatoirement un `Makefile` qui permettra de lancer la compilation facilement. Le respect des fonctionnalités imposées et la qualité du code (structure, utilisation **pertinente** d'un maximum de concepts - héritage, polymorphisme etc. - vus en cours et en TP, commentaires) seront primordiaux.

On sépare les besoins d'architecture du code, des besoins fonctionnels.

Infrastructure du projet :

- Un fichier `Makefile` pour compiler votre programme ;
- Chaque classe est dans un fichier à part, et a son fichier de header ;
- Un fichier `main.cpp` qui lance le jeu ;
- Dans le rapport un guide d'utilisation de la compilation, notamment s'il y a plusieurs modes possibles

Fonctionnalités du jeu :

- La base du jeu fonctionnelle :
  - La possibilité de jouer à 2 joueurs en initialisant correctement les cartes.
  - Une initialisation correcte des decks de base (7 *Or*, 1 *Épée courte*, 1 *Dague*, 1 *Rubis*), et des points de vie (50).
  - Les cartes de tout le `base set` pour le marché (dont certaines en plusieurs exemplaires comme dans le jeu officiel).
  - La possibilité de jouer les tours : acheter les cartes du marché, jouer des cartes,
  - Tous les effets : capacités *alliés*, capacités *activer*, effets de *sacrifice* qui peuvent être activés au moment adapté (à la pose ou ultérieurement dans le tour).
  - L'attaque : attaque de l'adversaire, des champions, des gardes en priorité.
- Un God-mode que l'on peut activer et désactiver **à tout moment** depuis une partie classique qui permet :
  - De mettre chaque joueur à un point de vie
  - D'acheter des cartes depuis toute la pioche du marché, et pas seulement les cinq quartes visibles du marché.
  - De plus une fois achetées, elles arrivent dans la main directement.

L'affichage pourra être assez minimaliste sur la console.

3.4 Extensions supplémentaires

Nous proposons dans cette section plusieurs possibilités d'améliorations, de niveaux de difficultés variables, et qui chacune peut mettre en place différentes notions que vous avez apprises dans ce module.

Pour chaque extension proposée, on vous demandera dans le rapport de la décrire succinctement et la façon de la réaliser.

Voici ici un résumé de quelques possibilités d'extensions, avec une estimation de leur niveau de difficulté, mais sans ordre de préférence :

Nom	**Descriptif et idées**	**Difficulté**
IA	Créer une IA contre laquelle jouer, à 2, 3 ou 4 joueurs potentiellement. L'IA peut être assez complexe et prendre en compte tous les mécanismes du jeu	⋆ à ⋆⋆⋆
Interface graphique	Log des actions, couleur de console, affichage d'une forme de terrain de jeu (ascii art, rafraichissement console), utilisation bibliothèque graphique, etc.	⋆ à ⋆⋆⋆
Plus de deux joueurs	Possibilité de jouer à plus de 2 joueurs, en console ou en ligne	⋆ à ⋆⋆⋆
En ligne	Possibilité de jouer à distance	⋆⋆⋆
Sauvegarde de partie	Possibilité de s'arrêter en cours de partie, et de revenir	⋆⋆
Extensions du jeu	Possibilité d'ajouter des sets de cartes supplémentaires de toutes les extensions du jeu (cf. sur le site de jeu et site officiel)	⋆ à ⋆⋆⋆