



GLIN403 – Projet noté « Dés Primés »

Programmation Impérative 2
Alban MANCHERON

L'objectif de ce projet est la réalisation d'un jeu de dé (à 6 faces) en C++. Avant de définir les attentes pédagogiques, il convient donc de présenter les...

Règles du jeu









Le jeu peut se jouer à 3 joueurs ou plus¹. Chaque joueur dispose au début de la partie de 5 dés (ce nombre est fluctuant au cours de la partie).

Avant de démarrer la partie, chaque joueur lance les dés, celui qui totalise le plus grand score commence (en cas d'égalité de score maximum, ce processus est répété entre les joueurs concernés, jusqu'à ce qu'il n'y ait plus qu'un lauréat). Le sens de rotation est alors défini en fonction du voisin du joueur qui commence ayant le score le plus élevé (en cas d'égalité, le sens de rotation est celui des aiguilles d'une montre).

Exemple 1 *Considérons une partie à 6 joueurs : J_1, \dots, J_6 . Chaque joueur lance ses 5 dés. Les joueurs obtiennent respectivement 8, 19, 21, 17, 21, 21. Les trois joueurs J_3 et J_5 et J_6 sont à égalité. Ils relancent tous les trois leurs dés et obtiennent respectivement 12, 18 et 13. C'est donc au joueur J_5 de commencer. Comme le joueur J_6 avait obtenu 21 en première instance, c'est donc lui qui sera le prochain joueur, puis J_1, J_2, \dots*

La partie se termine dès lors que l'un des joueurs n'a plus de dé. Le vainqueur est le joueur qui a le score le plus élevé.

Les points et actions modifiant le cours de la partie sont définis ainsi :

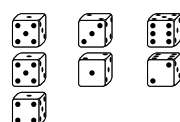
1. Le joueur courant lance tous ses dés.
2. Si le joueur a plus de 3 dés **et** qu'il n'obtient que des , le joueur marque 3 points par dé lancé et la partie est terminée. Dans le cas contraire, les  sont ignorés.
3. Chaque  donne 1 point au joueur courant.
4. Chaque  donne 2 points au joueur courant et 1 point à ses deux voisins.
5. Si le joueur a un nombre impair de , le sens du jeu est inversé. De plus, chaque  fait passer son tour au joueur qui aurait dû jouer ensuite (le joueur courant lui-même peut-être concerné).
6. Il donne ensuite ses  à son voisin ayant le plus petit score (en cas d'égalité, les dés sont donnés au voisin qui a joué en dernier).
7. Il se défause de tous ses  (les dés sont retirés du jeu).
8. C'est au joueur suivant de jouer.

Exemple 2 *Considérons une partie à 6 joueurs : J_1, \dots, J_6 . Le joueur J_1 vient de jouer et c'est à J_2 de lancer ses dés, le prochain joueur à jouer est a priori J_3 , sauf que celui-ci doit passer son tour, ce sera donc normalement à J_4 de jouer au prochain tour. Voici l'état du jeu avant et après le lancé de dés par J_2 :*

1. À 6, ça devient vraiment sympathique...

Joueur	#dés	score
J_1	3	5
J_2	7	3
J_3	4	6
J_4	2	4
J_5	3	3
J_6	5	4

J_2 lance ses dés



Joueur	#dés	score
J_1	5	6
J_2	4	6
J_3	4	7
J_4	2	4
J_5	3	3
J_6	5	4

En effet, le est ignoré. Le lui donne 1 point (J_2 passe donc à 4 points). Le lui donne 2 points et 1 point à ses voisins (les scores respectifs de J_1 , J_2 et J_3 passent donc à 6, 6 et 7 points). Il y a 1 seul . Donc le sens du jeu est inversé et le joueur J_1 passe son tour (ce sera donc au joueur J_6 de jouer le prochain tour). Le joueur J_1 ayant le moins de points, il récupère les deux . Enfin le est retiré du jeu.

Travail à réaliser

Vous devez implémenter ce jeu en C++, en utilisant les concepts abordés pendant l'ensemble des cours, TD, TP. Pour cela, il vous est demandé d'établir dans un premier temps le cahier des charges du projet (e.g., diagrammes UML). Une fois cette analyse réalisée, vous pourrez commencer à développer votre programme.

Parmi les fonctionnalités que vous aurez à mettre en place, vous devrez :

- permettre de jouer une partie à plusieurs (chaque joueur est identifié par un pseudo).
- permettre de jouer contre des joueurs virtuels.
- enregistrer le journal d'une partie (son déroulement, l'ensemble des actions réalisées).
- maintenir le top 10 des scores réalisés par un joueur réel uniquement.
- définir un mode *debug* permettant de dérouler une partie pas à pas et au besoin de changer la configuration de la partie courante (pratique pour vérifier/corriger son programme).

Vous veillerez à coder proprement (respect des normes ANSI, commentaires utiles, indentation cohérente, fichiers structurés², ...).

Vous veillerez également à réaliser des tests unitaires pour chaque classe.

Votre programme devra bien évidemment être utilisable en ligne de commande et disposer d'une option `--help` décrivant comment l'exécuter.

S'il vous reste du temps, de l'énergie et de la motivation, vous pouvez (mais ce n'est en rien obligatoire) définir une interface graphique dans le langage de votre choix pour rendre votre jeu plus agréable (e.g., librairie *wxWidgets*, application mobile via NDK, TCL/TK, ...).



Soyez toutefois assuré que cette dernière fonctionnalité ne sera prise en compte que très faiblement dans le calcul de la note et seulement dans l'hypothèse où le travail demandé aura été correctement réalisé.

Enfin, vous devrez rendre sur l'ENT – au plus tard le 30 avril 2014 – une archive compressée (au format `tar.gz` ou `tar.bz2` uniquement) contenant³ :

- les sources de votre programme ;
- un fichier texte⁴ mentionnant vos noms et prénoms ;
- un fichier texte⁵ expliquant comment compiler (en ligne de commande) et utiliser votre programme⁶ ;
- un rapport⁷ au format **pdf** mentionnant vos noms et prénoms et décrivant votre analyse, vos choix, vos tests, les remarques et commentaires, ...

Concernant votre organisation, le travail est à réaliser en binôme (pas de trinôme, quadrinôme, ...); le travail en monôme est toléré.

Bon Courage. . .

2. La règle 1 classe = 1 .h + 1 .cpp est raisonnable.

3. Le non-respect de ces consignes sera indubitablement sanctionné.

4. Typiquement, un fichier `Auteurs.txt`.

5. Typiquement, un fichier `LisezMoi.txt`.

6. L'utilisation d'un `Makefile` est conseillée.

7. Le rapport devra être écrit en français et respecter au mieux les règles typographiques françaises.