V1.0 Projet CY-Fish

CLASSE préING1 • 2023-2024
AUTEURS Eva ANSERMIN - Romuald GRIGNON
E-MAILS eva.ansermin@cyu.fr - romuald.grignon@cyu.fr

DESCRIPTION DU PROJET

Le but de ce projet est de réaliser un petit jeu sur le terminal où des pingouins sur un iceberg s'affrontent pour être celui qui attrapera le plus de poissons.
Le jeu se déroule sur une grille **hexagonale** sur laquelle se trouvent des cases qui contiennent des poissons (1 ou plusieurs poissons par case) et des pingouins (plusieurs par joueur) qui se déplacent sur les cases.
Chacun leur tour, les joueurs vont déplacer **un** de leurs pingouins sur la grille pour atteindre une autre case. Après le mouvement, la case de départ sur laquelle était le pingouin avant de se déplacer est collectée par le joueur (il récupère les poissons qui étaient sur cette case). Cette case n'est plus utilisable, l'iceberg a fondu à cet endroit.
Un pingouin ne peut se déplacer qu'en **ligne droite**, d'autant de cases que le joueur souhaite et autant que la configuration le permet. Un pingouin ne peut pas traverser une case occupée par un autre pingouin, et ne peut pas traverser une case qui a été retirée du plateau de jeu. Une case ne peut être occupée que par un seul pingouin à la fois.
Quand un joueur ne peut plus jouer (il n'a plus la possibilité de déplacer l'un de ses pingouins), alors c'est au joueur suivant de jouer tout simplement.
Quand tous les joueurs sont bloqués, la partie se termine et les comptes sont faits pour déterminer le joueur vainqueur.

INFORMATIONS GENERALES

Taille de l'équipe
Ce projet est un travail d'équipe. Il est autorisé de se réunir en groupe de 3 au maximum. Si le nombre total d'étudiants n'est pas un multiple de 3 et/ou si des étudiants n'arrivent pas à constituer des groupes, c'est au chargé de TD de statuer sur la formation des groupes. Pensez-donc à anticiper la constitution de vos groupes pour éviter des décisions malheureuses.

Démarrage du projet
Vous obtiendrez de plus amples informations quant aux dates précises de rendu, de soutenance, les critères d'évaluation, le contenu du livrable, …, quand le projet démarrera officiellement.

Dépôt de code

Vous devrez déposer la totalité des fichiers de votre projet sur un dépôt central Git. Il en existe plusieurs disponibles gratuitement sur des sites web comme github.com ou gitlab.com.

Rapport du projet

Un rapport écrit est requis, contenant une brève description de l'équipe et du sujet. Il décrira les différents problèmes rencontrés, les solutions apportées et les résultats. L'idée principale est de montrer comment l'équipe s'est organisée, et quel était le flux de travail appliqué pour atteindre les objectifs du cahier des charges. Le rapport du projet peut être rédigé en français.

Démonstration

Le jour de la présentation de votre projet, votre code sera exécuté sur la machine de votre chargé(e) de TD. La version utilisée sera la dernière fournie sur le dépôt Git avant la date de rendu. Même si vous avez une nouvelle version qui corrige des erreurs ou implémente de nouvelles fonctionnalités le jour de la démonstration, c'est bien la version du rendu qui sera utilisée.

Organisation

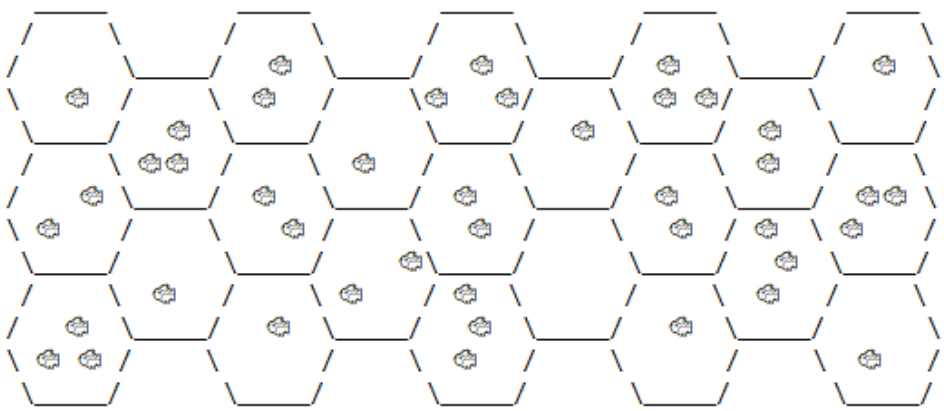
Votre projet complet **devrait (dans l'idéal)** être stocké sur un dépôt git (ou un outil similaire) tout au long du projet pour au moins trois raisons : éviter de perdre du travail tout au long du développement de votre application, être capable de travailler en équipe efficacement, et partager vos progrès de développement facilement avec votre chargé de projet. De plus il est **recommandé** de mettre en place un environnement de travail en équipe en utilisant divers outils pour cela (Slack, Trello, Discord, …).

CRITERES GENERAUX

- Le **but principal** du projet est de fournir une **application fonctionnelle** pour l'utilisateur. Le programme doit correspondre à la description en début de document et implémenter toutes les fonctionnalités listées.
- Votre code sera généreusement **commenté**.
- Tous les éléments de **votre code** (variables, fonctions, commentaires) seront écrits dans la **même langue**. Langue anglaise conseillée mais pas obligatoire.
- Votre application ne doit **jamais** s'interrompre de manière intempestive (crash), ou tourner en boucle indéfiniment, quelle que soit la raison. Toutes les erreurs doivent être gérées correctement. Il est préférable de d'avoir une application stable avec moins de fonctionnalités plutôt qu'une application contenant toutes les exigences du cahier des charges mais qui plante trop souvent. Une application qui se stoppe de manière imprévue à cause d'une erreur de segmentation ou d'une exception, par exemple, sera un événement très pénalisant.
- Votre application devra être **modulée** afin de ne pas avoir l'ensemble du code dans un seul et même fichier par exemple. Apportez du soin à la conception de votre projet avant de vous lancer dans le code.
- Le livrable fourni à votre chargé(e) de TD sera simplement l'**URL** de votre **dépôt Git** accessible **publiquement**. Même si vous n'avez pas utilisé ce dépôt régulièrement au cours du projet, le code final sera livré dessus.

FONCTIONNALITES DU PROJET

- Au début du jeu, le programme demande le nombre de joueurs qui vont participer au jeu. Le nombre possible de joueurs est compris entre 2 et 6. Le programme demandera les noms de chaque joueur pour pouvoir les afficher.
- Le jeu doit également fixer la taille du plateau de jeu (nombre de lignes et nombre de colonnes).
- Le nombre de pingouins est fonction du nombre de joueurs : pour 2 joueurs, il y a 4 pingouins par joueur sur le plateau. Pour 3 joueurs il y a 3 pingouins par joueur, et pour 4 joueurs il y a 2 pingouins par joueur.
- L'affichage doit comporter la zone de jeu rectangulaire ou carrée avec les cases de forme hexagonale. Le nombre de colonnes et de lignes doit être au moins de 9 colonnes et 9 lignes. Votre modèle de données (la zone mémoire qui contient votre grille de jeu) peut, lui, être une grille orthogonale (une matrice) : il vous faudra alors effectuer une petite conversion entre les coordonnées orthogonales et hexagonales, ainsi que les mouvements possibles. Voici un exemple possible d'affichage de cases hexagonales dans un terminal (9 colonnes sur 3 lignes) :



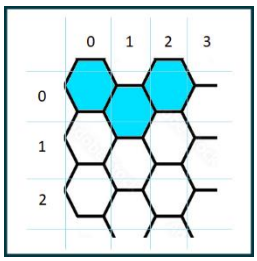
- L'affichage doit toujours afficher les noms et scores de chaque joueur en plus de la zone de jeu.
- Chaque case hexagonale doit comporter entre 1 et 3 poissons, valeur fixée aléatoirement par le programme au démarrage. Si un pingouin se trouve sur une case, il faut également pouvoir l'identifier (savoir à quel joueur appartient ce pingouin). Le programme doit permettre de vérifier qu'il y a suffisamment de cases avec 1 seul poisson car les pingouins démarrent toujours sur une case avec 1 poisson.
- Le nombre de poissons sur chaque case doit être visible.
- A chaque tour, le joueur actif DOIT déplacer un de ses pingouins d'au moins 1 case.
- A chaque tour, le joueur actif doit choisir quel pingouin déplacer, puis doit indiquer la direction de déplacement (parmi 6 puisque nous sommes sur une grille hexagonale) puis il doit indiquer la distance de déplacement (le nombre de cases). Ensuite le programme doit vérifier si ce déplacement est valide (le pingouin ne peut pas se déplacer sur une case déjà occupée par un autre pingouin, il ne peut pas se déplacer en dehors du plateau de jeu, et ne peut pas se déplacer sur une case vide. Il ne peut pas non plus traverser ces cases pour s'arrêter derrière).
- Si le déplacement n'est pas valide, le programme redemande au même joueur d'indiquer à nouveau son déplacement.
- Le programme vérifie toujours qu'il y a au moins 1 déplacement possible.

- Si un joueur n'a aucun déplacement possible, son tour est passé automatiquement par le programme.
- Quand un déplacement est valide, le pingouin impacté est téléporté sur la case de destination et la case de départ est supprimée (il n'est plus possible de se déplacer dessus, ni de la traverser). Le nombre de poissons de la case supprimée est ajouté au score du joueur courant.
- Les cases supprimées doivent être facilement identifiable à l'écran.
- Quand il n'y a plus aucun joueur qui peut effectuer de déplacement, c'est le critère pour marquer la fin du jeu. Un message avec le nom du gagnant est affiché, ainsi que les scores. Le programme doit permettre de recommencer une partie ou quitter, en fonction du choix de l'utilisateur.

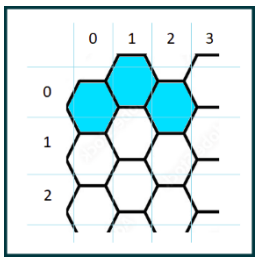
VARIANTES

- **Grilles Hexagonales**

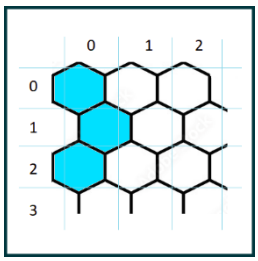
  Ci-dessous se trouvent les 4 variantes de grilles hexagonales possibles. Les cases en bleues représentent ici la première ligne (hexa A et B) ou la première colonne (hexa C et D) du tableau 2D en mémoire.



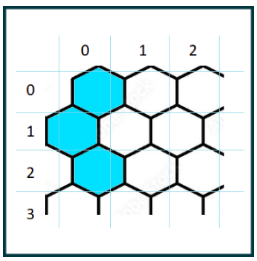
variante **HEXA_A**



variante **HEXA_B**



variante **HEXA_C**



variante **HEXA_D**

- **Positions de départ des pingouins**

  Lors du début de partie, les positions initiales des pingouins peuvent être choisies au hasard par le programme, ou bien manuellement par chaque joueur à tour de rôle. Il y a donc 2 variantes pour cette fonctionnalité.

  Attention, un joueur ne peut choisir qu'une case qui possède 1 seul poisson.

  Le programme doit toujours vérifier cette condition, quelle que soit la variante imposée. Cela veut dire aussi que lors de la génération de la carte il doit y avoir assez de cases avec 1 poisson pour accueillir tous les joueurs en début de partie.

  variante **START_AUTO** : positions choisies par le programme
  variante **START_MANUAL** : positions choisies par les joueurs

- **Comptage des poissons**
  Cette variante amène quelques modifications sur la règle de base qui dit que chaque poisson vaut 1 point. Les cases de départ sont toujours des cases avec 1 poisson qui ne vaut qu'un seul point quelle que soit la variante.

  variante **FISH_NORMAL** : Les poissons valent tous 1 point. Règle de base.
  variante **FISH_ROTTEN** : Les poissons valent tous 1 point, sauf que parfois il y a des poissons avariés qui se cachent parmi les poissons de la case. Ces poissons avariés valent -1 point. Un message s'affichera quand un pingouin ramassera un poisson avarié. Il ne peut y avoir au maximum qu'un seul poisson avarié parmi une case qui en contient 2 ou 3. Les poissons seuls sur une case sont toujours propres à la consommation. Les poissons avariés sont insérés aléatoirement par le programme.
  variante **FISH_GOLDEN** : chaque poisson vaut 1 ou 2 ou 3 points. Il faut trouver le moyen de bien les identifier graphiquement à l'écran. Leur nombre ne change pas par rapport aux règles de base. Les poissons qui sont seuls sur leurs cases peuvent valoir plus que 1 point mais le programme doit garantir qu'il y a suffisamment de poissons seuls valant 1 point pour servir de cases de départ du jeu.

RESSOURCES UTILES

Github
- www.github.com
- https://docs.github.com/en/get-started/quickstart/hello-world

Modification des couleurs (police et fond) dans le terminal
- http://sdz.tdct.org/sdz/des-couleurs-dans-la-console-linux.html

Affichage d'emoji dans le terminal
- Liste des emojis : https://unicode.org/emoji/charts/full-emoji-list.html