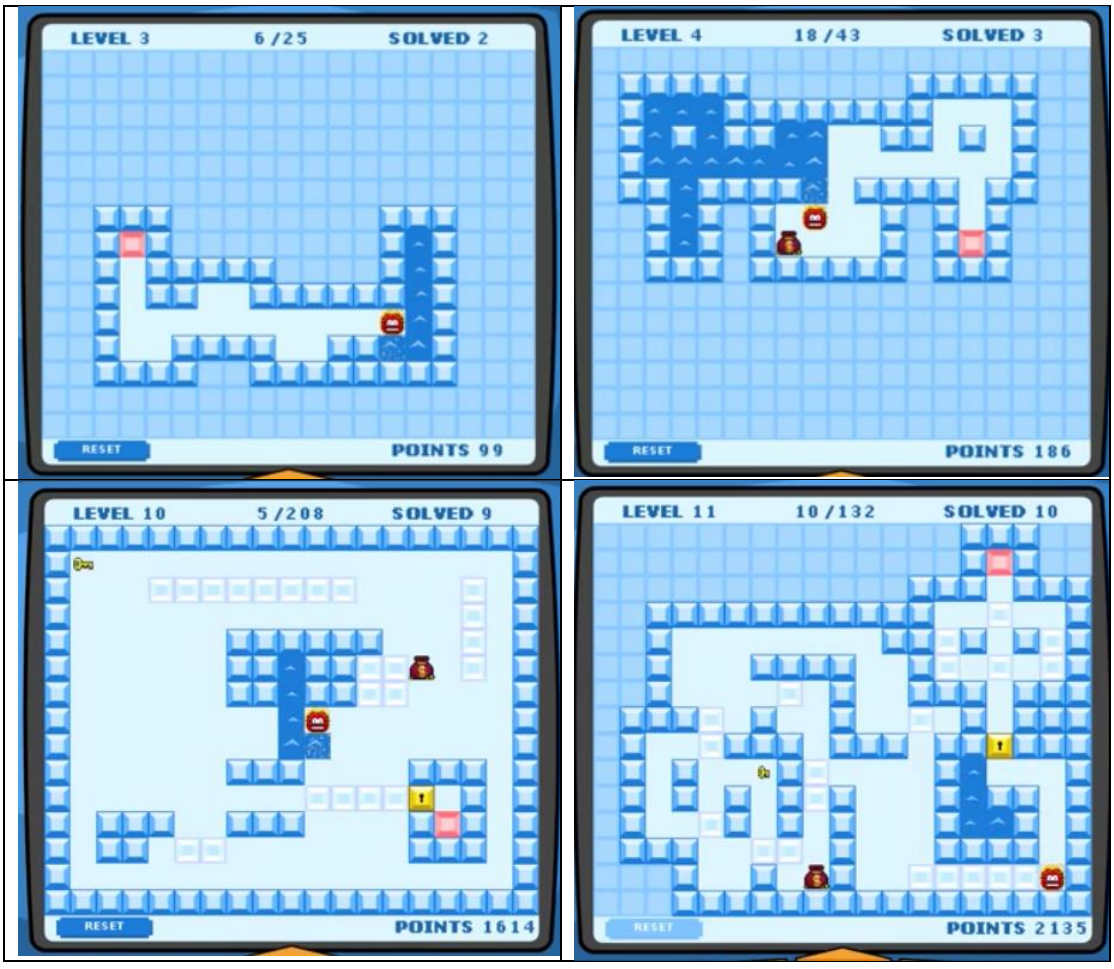
FINN-  ECE


LEVEL 3 6/25 SOLVED 2
RESET POINTS 99
LEVEL 4 18/43 SOLVED 3
RESET POINTS 186
LEVEL 10 5/208 SOLVED 9
RESET POINTS 1614
LEVEL 11 10/132 SOLVED 10
RESET POINTS 2135

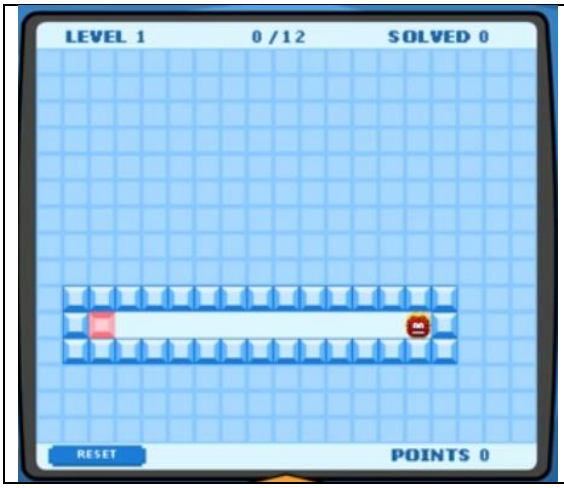
Objectif de ce projet

L'objectif de votre projet est de réaliser le jeu "Club Penguin Thin Ice", dans le langage orienté objet Java, en mode console, puis graphique.
Vous jeu devra permettre à un joueur de déplacer son personnage ECEMAN sur plusieurs tableaux, chacun composé de plusieurs blocs.

Pour passer plus facilement du mode console au mode graphique, vous devrez définir une organisation modulaire multi-fichiers respectant l'approche **Modèle-Vue-Contrôleur :** Modèle Vue Contrôleur (wikipedia) et Adopter une architecture MVC (openclassrooms). Votre diagramme de classes devra montrer ce découpage modulaire : encadrer avec 3 couleurs différentes les classes faisant partie du Modèle (couleur 1), de la Vue (couleur 2) ou du Contrôleur (couleur 3)

Présentation du jeu

Inspiré du jeu "Club Penguin Thin Ice", le jeu que nous vous proposons comme projet informatique de ce semestre est un jeu d'arcade à réaliser en mode console, puis graphique. Les illustrations ci-dessous de la figure 1 montre ce jeu en mode graphique (à gauche) et en mode console (à droite).




```
####################
####################
####################
####################
####################
####################
####################
#MMMMMMMMMMM#
#MPooooooooooooEM#
#MMMMMMMMMMM#
####################
```

Figure 1

Dans ce jeu, le personnage principal, ECEMAN, se déplace dans les 4 directions dans un univers composé de glace fine, de murs, d'obstacles, de banquise épaisse, de tunnels, d'outils, de bonus, d'une porte de sortie et... d'ennemis. En mode console, les blocs de glace fine, de murs, … ennemis seront chacun représenté par un caractère.

À chaque fois qu'ECEMAN passe sur une case de glace, celle-ci se brise pour faire apparaître de l'eau et empêcher tout passage deux fois sur la même case. ECEMAN doit passer par toutes les cases de glace, fine (se brise au premier passage dessus) ou épaisse (se brise au deuxième passage dessus), et terminer par la porte pour gagner le tableau suivant.

Vous trouverez sur youtube des vidéos, comme ci-dessous, illustrant le fonctionnement des premiers tableaux de ce jeu qui pourront vous aider à comprendre et vous servir d'inspiration.

Club Penguin Thin Ice Complete Walkthrough - YouTube

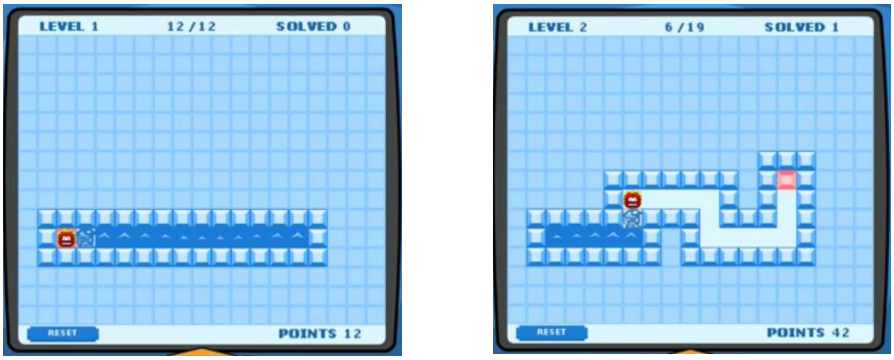


Figure 2

Cahier des charges du déroulement du jeu

1) Jeu en mode console (sur 16 points)

I. Le menu d'accueil

Le jeu débute par un menu utilisateur permettant au joueur :

- de prendre connaissance des règles du jeu et touches du clavier à utiliser
- de débuter une nouvelle partie,
- de continuer une partie sauvegardée,
- de voir les scores,
- …

II. Caractéristiques du jeu

Le plateau de jeu se présente comme une matrice de caractères 15 lignes x 19 colonnes contenant des caractères choisis pour représenter les différents éléments du jeu (Eceman, murs, glace…). Ces caractères peuvent être semi-graphiques et en couleurs : voir les liens **Table Unicode des caractères semi-graphiques**
et **La gestion des couleurs en Java sur Windows**
dans la section Ressources externes

III. 5 tableaux à réaliser

Vous réaliserez au minimum 5 tableaux de difficultés croissantes.
Figure 1

i. Le premier tableau devra contenir uniquement de la glace fine et des murs (ils ne bougent pas) et permettre au joueur la prise en main du personnage principal : voir Figure 1

ii. Le deuxième tableau ajoute la « banquise épaisse » qui sont en fait des cases sur lesquelles ECEMAN peut passer deux fois avant qu'elles ne se brisent (au premier passage, la banquise épaisse se transforme en glace fine)

iii. Le troisième tableau met en scène des outils comme par exemple :

- « une tondeuse » qu'ECEMAN peut pousser et qui brise toute la glace sur sa lancée jusqu'à s'arrêter sur un mur.
- Des briques de mur qu'ECEMAN peut pousser pour les déplacer dans les 4 directions
- Des « potions de légèreté » qui permettent à ECEMAN de passer sur la glace sans la briser

iv. Le quatrième tableau ajoute un tunnel dans lequel ECEMAN doit passer pour passer d'une zone fermée du tableau à une autre zone fermée qui contient entre autres la porte de sortie (le tunnel est virtuel, la porte d'entrée du tunnel se trouve dans une des zones fermées et lorsqu'ECEMAN l'emprunte il ressort par l'autre porte du tunnel dans une autre zone fermée, sans possibilité de faire marche arrière).

v. Le cinquième tableau fera apparaitre, en plus, un ou plusieurs ennemis qui « balayent le terrain » en se déplaçant de gauche à droite ou de haut en bas en changeant de direction au contact d'un mur ou d'un obstacle. Si ECEMAN touche un ennemi, le tableau est perdu et doit être recommencé.

Tous les tableaux doivent être jouables. Aucune case de glace ne doit être inaccessible à ECEMAN (par exemple, complètement entourée de murs) et l'empêcher de terminer un tableau. Tout tableau terminé est sauvegardé dans un fichier pour permettre de continuer une partie sauvegardée (tableau suivant), à partir du menu : voir la section **Le menu d'accueil**

Extensions possibles sur les tableaux (bonus de 2 points)

- Tableaux supplémentaires apportant des innovations ou des difficultés spécifiques (nouveaux outils, nouveaux bonus, nouveaux ennemis…)
- Editeur de tableau
- Musique …

IV. Déroulement d'une partie

Lorsque l'utilisateur commence à jouer (nouvelle partie ou reprise d'une ancienne partie) le tableau affiché est à refaire depuis le début.

Il peut déplacer le curseur à l'aide des touches 2, 4, 6 et 8 du pavé numérique (pad), ou des touches 'Z' 'Q' 'S' et D s'il n'a pas de pad, pour faire bouger ECEMAN d'une case dans les 4 directions. Pour déplacer en Java un personnage, voir le lien **Déplacer en Java un personnage sur une map** dans la section Ressources externes

Si ECEMAN passe sur un objet bonus, il « consomme » cet objet et change son comportement en conséquence. S'il bute sur un obstacle, il se bloque. S'il touche un objet mobile, il le pousse d'une case.

V. Score stocké dans un fichier

Un compteur de temps et de cases de glace brisées par niveau servira de score et sera stocké dans un fichier lorsque le joueur quitte la partie. Tout niveau non terminé n'est pas pris en compte dans les scores.

2) Jeu en mode graphique avec Listeners (4 points)

Dans ce mode graphique, il vous fera reprendre vos méthode d'affichage dans le module de *Vue*. De même dans le module *Contrôleur*, **vous devrez faire en sorte que l'utilisateur puisse déplacer ECEMAN avec la souris**, grâce à l'utilisation de *Listeners* (écouteurs).

Planning et organisation

Période de réalisation du projet : de la semaine du 11 octobre à la semaine du 6 décembre 2021 incluse. Des soutenances auront lieu la semaine du 13 décembre 2021.

Équipes : en trinôme ou éventuellement en binôme dans un même groupe de TD.

Evaluation : La version finale sera présentée à la soutenance du projet qui donnera lieu à la note de projet.

Un diagramme de classes doit être présenté pour montrer la conception objet, y compris avec héritage, en respect du travail demandé, contraintes et consignes ci-dessous.

Versionning de votre projet : GIT + Bitbucket

Vous pourrez utiliser l'outil de **versioning GIT** pour partager votre code de ce projet : voir le lien **Versioning avec Git** dans la section Projet Java 2021 : FINN-ECE de la page campus Cours : POO Java (app) (campusonline.me).

Lors de la soutenance, vous montrerez les différentes phases de développement de votre projet (les branches des "versions" du projet). Vous pourrez ainsi être en mesure de montrer "qui a fait quoi" dans le projet.
Grâce au versioning, vous n'aurez plus de problèmes et d'excuses du type :

- c'est mon camarade qui a tout le projet, je n'ai pas pu corriger les erreurs...
- mon disque dur est mort la veille de la soutenance...
- on m'a volé mon ordinateur le jour de la soutenance...
- j'ai renversé du liquide sur mon clavier...
- je ne comprends pas, mon code a été écrasé ? et je n'ai pas fait de backups...

Les grandes étapes à respecter

1- Analyse et conception générale du diagramme de classes.

- A partir du cahier des charges (CDC) : extraire les données pertinentes, les regrouper en grandes entités / objets, spécifier et caractériser les attributs et les fonctionnalités de chaque objet, identifier les interactions entre les différents objets ainsi que les différents scénarios possibles, …
- En déduire le diagramme de classes, mettant en relation les classes en y intégrant si possible de l'héritage et du polymorphisme. Pour chaque classe, définir les attributs (en général *private*, ou *protected* en cas d'héritage), et les méthodes (inutile d'y mentionner, les constructeurs, les getters et les setters).
- IHM : lister les choix à offrir au démarrage, lister les évènements à gérer, déterminer l'organisation et le contenu de l'écran de jeu (maquette), …
- Rédiger une Analyse Chronologique Descendante (ACD) du programme principal.
- Répartir les tâches au sein de l'équipe.

2- Analyse et conception détaillée

- Pour chaque classe, lister les prototypes de toutes les méthodes requises en précisant ses paramètres d'entrée et de sortie.
- Réaliser progressivement une maquette du jeu en testant au fur et à mesure du codage et en tenant compte des différents scénarios.
- Entre autres critères de qualité, le programme final devra être très facilement adaptable par tout autre développeur (exemples : changement des valeurs d'initialisation, changement des caractères et couleurs d'affichage, …).

3- Développement dans un langage objet (codage, tests).

Dans le langage objet **Java**, implémenter le jeu en respect de votre analyse des 2 étapes précédentes : **diagramme de de classes**, organisation modulaire multi-fichiers selon l'approche **Modèle-Vue-Contrôleur**, avec **héritage et polymorphisme**, commenter les prototypes des méthodes (fonctionnalités) en précisant les paramètres d'entrée/sortie et commenter aussi l'ACD de ces méthodes et du programme principal.

4- Critères de notation.

Vous devez réaliser l'ensemble du cahier des charges de base (expliqué dans la paragraphe précédent). Le langage de programmation doit être le langage objet Java avec héritage... Votre code devra être modulaire, respecter les interfaces en mode graphique et bien commenté !

Un code qui ne compile pas ou qui plante au démarrage ne vaut pas plus de 10/20. Tester donc votre programme avant de le déposer sur Campus...

Votre travail sera jugé sur les critères suivants :

- Le respect rigoureux des règles du jeu énoncé précédemment (CDC)
- La modularité de votre conception et donc de votre code
- La bonne répartition des tâches entre les membres de l'équipe
- L'intérêt, l'originalité, la jouabilité et toutes les caractéristiques que vous prendrez soin de mettre en avant lors de la soutenance.

5- Deadline du livrable à déposer sur campus.

La date et toutes les consignes sur le livrable à déposer sont spécifiées dans la section **Projet Java 2021 : FINN-ECE** de la page campus Cours : POO Java (app) (campusonline.me).

Le livrable avec **le PowerPoint et le code du jeu fonctionnel** est à rendre dans cette section sur le lien Version du livrable PowerPoint + code fonctionnel (version soutenance) : deadline le 12/12/2021 23h55. La réalisation et son bon fonctionnement sera présenté lors de votre soutenance.

6- Soutenance

Vous aurez 20 minutes par équipe pour montrer l'efficacité de votre travail. Prévoyez des « accès rapides » à vos différents tableaux pour ne pas avoir à faire tout le jeu pour en découvrir les spécificités des tableaux les plus lointains.

Ressources externes

I. Table Unicode des caractères semi-graphiques

La table des caractères Unicode ci-dessous vous permet d'afficher des caractères semi-graphiques à l'écran.

Exemple en Java pour afficher un cœur :

```
System.out.println("\u2764");
```

Source : java - Comment afficher les caractères spéciaux en java? (askcodez.com)

Exemple d'autres caractères Unicode semi-graphiques :

Source : Liste des caractères Unicode de la catégorie «Symbole, autre » (compart.com)

II. La gestion des couleurs en Java sur Windows

Exemple :

```
public class ColourConsoleDemo {
   public static void main(String[] args) {
      // TODO code application logic here
      System.out.println("\033[0m BLACK");
      System.out.println("\033[31m RED");
      System.out.println("\033[32m GREEN");
      System.out.println("\033[33m YELLOW");
      System.out.println("\033[34m BLUE");
      System.out.println("\033[35m Magenta");
      System.out.println("\033[36m CYAN");
      System.out.println("\033[37m WHITE");
   }
}
```

Source : java — Afficher la sortie System.out.println avec une autre couleur (it-swarm-fr.com)

III. Déplacer en Java un personnage sur une map

Source :
Déplacer un personnage sur une map - Java en mode Console par Lunaryos - OpenClassrooms