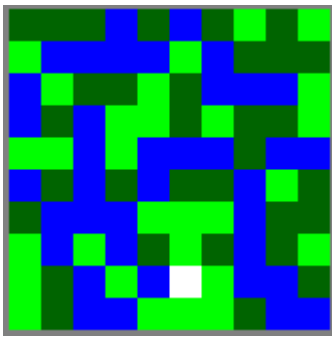
Jeu de rôle old-school



1. Types et fonctions de base

- Définir un type enregistrement *hero* contenant 3 niveaux (expérience, vie, attaque)
- Définir un type enum *typeTerrain* pouvant représenter de l'eau, de l'herbe ou un arbre
- Définir un type enregistrement *cell* contenant un type de terrain et un héros
- Créer une fonction *init* qui à partir d'une hauteur et d'une largeur, alloue une grille 2d de *cell* et la remplit avec des terrains de type aléatoire et un héros positionné aléatoirement
- Créer une fonction *affiche* qui affiche une grille dans la console, avec un entier entre 0 et 2 pour distinguer les terrains et un X pour le héros
- Créer une fonction *release* qui libère la mémoire occupée par une grille
- Tester la création et l'affichage d'une carte de 10x10 cellules

2. Affichage

Nous allons maintenant créer un affichage plus sympathique en utilisant le format Portable Pixmap (http://fr.wikipedia.org/wiki/Portable_pixmap). Cela consiste en l'écriture d'un fichier dans un format spécial où chaque case sera représentée par un pixel de couleur. Ce fichier pourra ensuite être interprété par un logiciel de lecture d'images.

Voici un résumé des informations contenues dans un tel fichier :

Each PPM image consists of the following:

1. A "magic number" for identifying the file type. A ppm image's magic number is the two characters "P3" (“P2” for pgm)
2. Whitespace (blanks, TABs, CRs, LFs).
3. A width, formatted as ASCII characters in decimal.
4. Whitespace.
5. A height, again in ASCII decimal.
6. Whitespace.
7. The maximum color value (Maxval), again in ASCII decimal. Must be less than 65536 and more than zero.
8. A single whitespace character (usually a newline).
9. A raster of Height rows, in order from top to bottom. Each row consists of Width pixels, in order from left to right. Each pixel is a triplet of red, green, and blue samples, in that order.

A row of an image is horizontal. A column is vertical. The pixels in the image are square and contiguous. Each sample in the raster is represented as an ASCII decimal number (of arbitrary size).

Each sample in the raster has white space before and after it. There must be at least one character of white space between any two samples, but there is no maximum. There is no

particular separation of one pixel from another -- just the required separation between the blue sample of one pixel from the red sample of the next pixel.
No line should be longer than 70 characters.
Strings starting with "#" may be comments.

Here is an example of a small image in this format.

```
P3
# feep.ppm
4 4
15
 0  0  0    0  0  0    0  0  0   15  0 15
 0  0  0    0 15  7    0  0  0    0  0  0
 0  0  0    0  0  0    0 15  7    0  0  0
15  0 15    0  0  0    0  0  0    0  0  0
```

There is a newline character at the end of each of these lines.

- Créer et tester une fonction *affichepgm* qui produit une image *map.pgm* à 4 niveaux de gris correspondant au contenu de la grille. Pour cela, la fonction devra :
  - ouvrir un fichier "map.pgm" (*fopen*)
  - écrire les 1eres lignes indispensables au fichier (*fprintf*)
  - parcourir les cellules de la grille et pour chacune écrire le code correspondant au contenu, en respectant la syntaxe ci-dessus
  - fermer le fichier (*fclose*)

- Sur le même principe, créer et tester une fonction *affichepppm* qui parcourt les cellules de la grille et pour chacune écrit le code RGB correspondant (bleu pour l'eau, vert clair pour l'herbe, vert foncé pour les arbres, blanc pour le héros) (définir une constante MAX_COLOR 255)

Après l'exécution de votre programme, on peut convertir une image Portable Pixmap. Ici, changer la taille par la commande console suivante et ouvrir le résultat avec firefox :

convert -scale 200x200 map.ppm map.gif

3. Animation

On cherche maintenant à déplacer le héros automatiquement et visualiser le résultat. Pour cela on créera une nouvelle image après chaque modification de la grille, et on rassemblera les différentes images en un *gif* animé

- Créer une fonction *deplaceHero* qui fait bouger le héros entre 2 *cell* données

- Créer une fonction *parcours* qui cherche le héros dans la grille, et le déplace 1 fois vers une cellule voisine ; cette case voisine sera choisie selon un algorithme d'aller-retour vers le bas de la grille : parcours d'une ligne dans une direction, puis passage à la ligne du dessous, parcours dans l'autre direction...

- Modifier la fonction d'affichage pour qu'elle prenne un identifiant en paramètre, qui servira à numéroter correctement chaque image : *map001.ppm, map002.ppm…* (*sprintf*)

- Dans le *main*, écrire et tester une boucle de simulation qui, tant que le héros peut se déplacer :
  - déplace le héros selon le *parcours*

- crée l'image correspondante
- incrémente le compteur d’images

Pour convertir ensuite dans la console les différentes images en une animation :

convert -loop 1 -delay 20 -scale 200x200 map.ppm map.gif*

- Faire toutes les modifications nécessaires pour pouvoir déplacer le héros sans le chercher dans la grille à chaque fois.

- Tester d’autres algorithmes de parcours

- Comment faire pour qu'il y ait 2 fois plus d'herbe que d'eau et d'arbres dans la grille ?