Projet d'info 4A

Wassim DJELLAT - Yann TROU

Avril 2020

Introduction

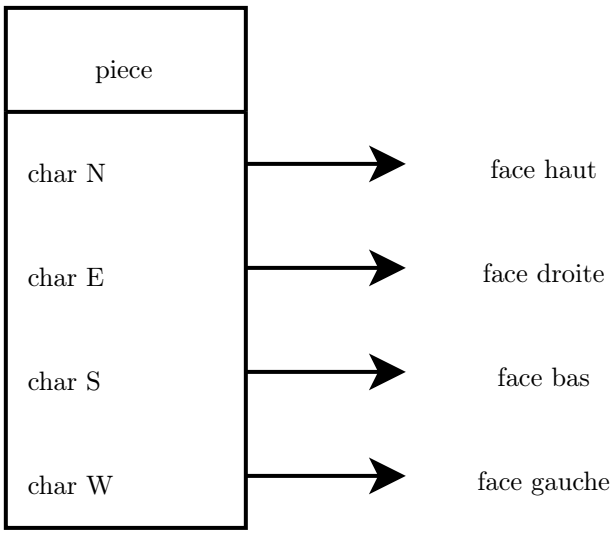
Variante choisie

Notre projet correspond à la version complexe, sans IA, avec la taille de tableau variable et la solution de départ aléatoire (7 ou I d'après les documents que vous avez envoyé).

I | mécaniques principales

1) Structures de données

Afin de modéliser le tableau de jeu contenant toutes les pièces, nous avons décidé d'utiliser la structure ci-contre :

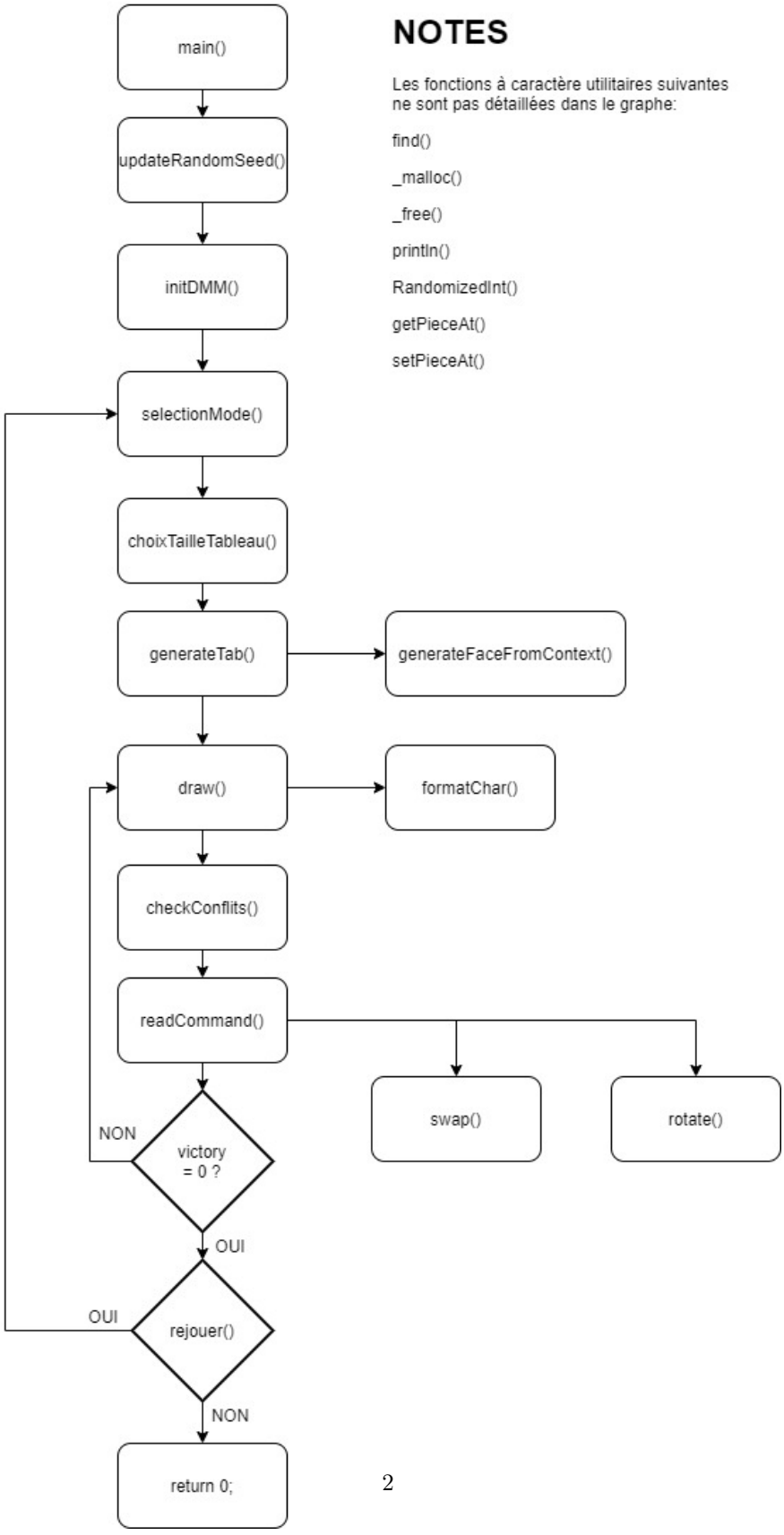


pour une pièce. Nous ne nous rappelons plus pourquoi avoir utilisé la notation géographique au lieu d'une notation plus lisible et évidemment nous ne le réalisons que pendant la rédaction du rapport.. Le tableau qui les contient, lui, est un tableau à une dimension car nous trouvions ceci plus facile qu'un tableau à deux dimensions. On utilise la formule ci dessous

$$x + cote * y = index$$

pour passer facilement de coordonnées 2D à coordonnées 1D.

2) Boucle de gameplay

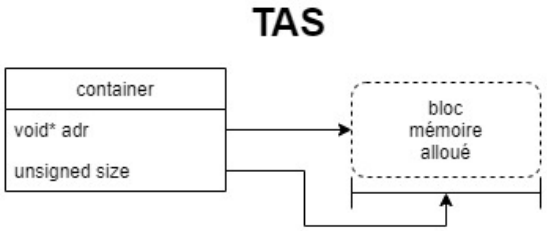


Le détail des fonctions est dans le code source joint. Ce graphe indique juste où sont appelées les fonctions principales, pas la structure des fonctions. La fonction initDMM() sert à initialiser la gestion de la mémoire dynamique, sinon fiez vous au nom pour l'utilité de la fonction.

II | Utilitaire

Gestion de la mémoire dynamique

Suite à des problèmes de fuite de mémoire dans formatChar(), et comme il y a avait un exercice similaire en TD, nous avons décidé de faire un système de gestion de la mémoire dynamique. Ainsi, nous avons une structure appelée container :



et un tableau index permettant l'indexation de tous les blocs mémoires. l'objectif final est de permettre la création et suppression de blocs mémoire n'importe ou dans le code sans problème.

Formatage du texte et affichage

Pour l'affichage, comme nous trouvions l'environnement du terminal fort austère, nous avons voulu mettre un peu de couleur grâce aux codes ANSI.