|  |
| --- |
| PROJET TOURS DE HANOI  \_  CAHIER DES CHARGES |
| Projet réalisé par :   * Solène Guillaume : [solene.guillaume@etu.univ-cotedazur.f](mailto:solene.guillaume@etu-univ-cotedazur.fr)r * Ambre Allouis : [ambre.allouis@etu.univ-cotedazur.fr](mailto:ambre.allouis@etu.univ-cotedazur.fr) |

Table des matières :

[**Présentation du jeu “Les tours de Hanoï"**](#_7rzuluyw1ttg) **1**

[**Objectifs du Projet**](#_c19qiiq3sfqu) **1**

[**Structure du Projet**](#_xtfaamrersqx) **1**

[Librairie](#_u0kjii8lbgez) 2

[Environnement](#_2fg7o2hijkrz) 2

[Le jeu](#_nbwqsxprj23m) 2

[Structure du code](#_5jp6pf7e7fw1) 2

# 

# Présentation du jeu “Les tours de Hanoï"

“Les tours de Hanoi” est un jeu classique pour enfants qui consiste à empiler des “donuts” sur des bâtons dans un ordre donné, tout en respectant certaines contraintes. Les règles du jeu sont simples :

* Initialement les donuts sont disposés de façon aléatoire sur toutes les tours sauf une seule ne disposant aucun anneau.
* Les anneaux ou “donuts” doivent être empilés du plus grand au plus petit.
* Chaque donuts est situé sur une tour, ils ne peuvent pas être en dehors d’une tour.
* On ne peut déplacer qu’un donuts à la fois.
* Chaque tour peut contenir au maximum le même nombre N d'anneaux.

Pour résumer, le jeu consiste à empiler et désempiler les anneaux jusqu’à les avoir placés dans le bon ordre sur chaque tour.

Pour la réalisation de ce projet, nous allons coder le jeu en C++ sous la forme d’un jeu dans un environnement informatique.

# Objectifs du Projet

* Création d’une application
* Création d’un jeu fonctionnel des tours de Hanoï
  + Création des tours
  + Création des anneaux
  + Déplacement des anneaux
  + Fin du jeu si succès
* Création d’un menu avec :
  + sous menu Jouer en solo
    - Niveau de difficulté facile (3 étages)
    - Niveau de difficulté moyen (4 étages)
    - Niveau de difficulté difficile (5 étages)
  + sous menu IA démonstration
    - Niveau de difficulté facile (3 étages)
    - Niveau de difficulté moyen (4 étages)
    - Niveau de difficulté difficile (5 étages)
  + Quitter
* Ajout d’un fond d’écran (menu et jeu)
* Mouvement des anneaux
* Ajout d’une musique

# Structure du Projet

Le projet se déroulera sur deux mois répartis en plusieurs séances de 4h chacune. Deux personnes travailleront dessus. Les tâches seront réparties équitablement entre elles.

Nous partons tout d’abord sur une interface graphique et nous rabatterons sur un graphisme écrit si l’on s'aperçoit que le modèle est difficile à appliquer dans le temps imparti.

## Librairie

Pour cela, nous pouvons utiliser la librairie SDL2 que nous avons déjà eu l’occasion d’utiliser l’année dernière en langage C. Cette librairie permet d’ouvrir une fenêtre, d’afficher des formes géographiques basiques et gérer des événements (clic de souris ou touche clavier).

D’autres librairies équivalentes sont intéressantes :

* Allegro
* SFML

A nous de déterminer par la suite laquelle est la plus simple d’utilisation.

## Environnement

Pour effectuer le projet, nous utiliserons un environnement Linux au travers d’une machine virtuelle. Cet environnement est optimal pour installer des libraires rapidement depuis le terminal de commandes. De plus, le projet sera facilement compilable sur tous les PC par lesquels il risque de transiter, que ce soit Windows ou Mac.

Afin de prévenir les éventuels crashs de la machine virtuelle, nous enregistrons régulièrement notre projet dans un repository GitHub trouvable sur le lien suivant : https://github.com/Solene49000/Projet\_Cpp

## Le jeu

Si le temps donné nous le permet, nous prévoyons de faire un menu qui donnera deux choix : résolution du jeu automatique par le PC ou bien partie lancée par l’utilisateur. La difficulté sera ensuite choisie entre trois niveaux :

* Facile : 3 anneaux
* Moyen : 4 anneaux
* Difficile : 5 anneaux

Ensuite le jeu se lancera. Trois tours seront affichées, les anneaux placés dans l’ordre sur la première tour. Le but sera de les mettre dans le même ordre sur la troisième tour.

Le jeu ne se terminera que lorsque la fenêtre se fermera ou que la partie est gagnée.

## Structure du code

Pour harmoniser la compilation, nous ferons un makefile afin de générer les fichiers .o correspondant à nos fichiers .hpp et .cpp.

Voici la liste préalable de nos fichiers :

* anneau.hpp
* anneau.cpp : constructeur + affichage
* tour.hpp
* tour.cpp : constructeur + affichage
* events.hpp
* events.cpp : constructeur
* main.cpp

Dans le tour.hpp, nous initialisons les variables globales :

* **N\_tour** = au nombre tours
* **N\_etage** = au nombre d’étages par tour
* **hauteur\_etage**  = Variable donnée
* **Largeur\_tour** = Variable donnée (nombre pair)
* **Hauteur\_tour** = N \* hauteur\_etage
* **Epaisseur\_tour** = Variable donnée (nombre pair)
* La structure : **tour\_structure** :
  + **numero\_tour** va de 1 à N\_tour
  + **nombre\_etage** = N\_etage
  + **position\_x** = Variable donnée
  + **position\_y** = Variable donnée
  + **position\_milieu\_x** = position\_x + (Epaisseur\_tour + Largeur\_tour)/2
  + **tab\_etage\_anneau[**nombre\_etage**]** ← contient les valeurs de 0 à 4 : 0 correspond à “absence d’anneaux” et de 1 à 4 pour le numéro de l’anneau

La page anneau.hpp contient la structure : **anneau\_structure** avec les données suivantes :

* **numero\_anneau** qui déterminera le numéro de l’anneau (plus l’anneau est gros, plus le numéro est grand)
* **hauteur\_anneau** = hauteur\_etage
* **largeur\_anneau** = numero\_anneau x largeur\_unitaire\_anneau
* **largeur\_unitaire\_anneau** : variable donnée
* **numero\_tour** qui indiquera sur quelle tour est positionné l’anneau
* **numero\_etage** qui indiquera à quel étage est positionné l’anneau

Pour afficher l’anneau, nous ferons appel à la classe tour afin de déterminer la position en x et y des anneaux.

Dans un premier temps, le déplacement des anneaux se fera par clic : clic sur tour 1 pour sélectionner l’anneau puis clic sur tour 2 pour supprimer l’anneau de la tour 1 et afficher l’anneau sur la tour 2.