|  |
| --- |
|  |
| DM Programmation C |
| Mots-mêlés |

|  |
| --- |
| Clément GAUDET / Maxime JAILLARD  19/11/2020 |

|  |
| --- |
| L2 – ProgC |
| Projet Programmation C |
| Threetogo |

Clément GAUDET / Maxime JAILLARD

16/01/2021

Sommaire

[Manuel utilisateur 3](#_Toc61604034)

[Listing des fonctions du programme 3](#_Toc61604035)

[token.c 3](#_Toc61604036)

[moteur.c 3](#_Toc61604037)

[graphique.c 4](#_Toc61604038)

[threetogo.c 4](#_Toc61604039)

[Architecture du site 5](#_Toc61604040)

[Arborescence des pages 5](#_Toc61604041)

[Requêtes SQL majeures 5](#_Toc61604042)

[Pistes d’amélioration 5](#_Toc61604043)

# Manuel utilisateur

/\*\* A COMPLETER \*\*/ (utilisation du makefile ? / commande de compilation ?)

Une fois le programme exécuté, la partie commence immédiatement et le timer se met en marche.

# Listing des fonctions du programme

## token.c

**Token \*alloc\_token(int color, int shape);**  
Effectue l'allocation mémoire d'un token et renvoie son adresse

**int length(Liste tokens);**  
Renvoie le nombre de tokens dans la liste

**Token \*next\_shape(Token \*token);**  
Renvoie le prochain token de la même forme dans la liste

**Token \*next\_color(Token \*token);**  
Renvoie le prochain token de la même couleur dans la liste

**void update\_shape\_links(Token \*token);**  
Met à jour le double chainage de la forme du token

**void update\_color\_links(Token \*token);**  
Met à jour le double chainage de la couleur du token

**void attach\_to\_head(Liste \*liste, Token \*newtok);**  
Attache un token en première position d'une liste

**void attach\_to\_tail(Liste \*liste, Token \*newtok);**  
Attache un token en dernière position d'une liste

**Token \*detach\_token(Liste \*liste, Token \*tok);**  
Détache le token de la liste

**int swap(Liste \*lst, Token \*tok1, Token \*tok2);**  
Echange tok1 et tok2 dans la liste lst. ATTENTION, NE CHANGE PAS LES DOUBLES CHAINAGES

## moteur.c

**int init\_queue(Liste \*queue);**  
Initialise une queue de 5 tokens générés aléatoirement sous forme de liste chainée

**void free\_liste(Liste \*liste);**  
Libère toute la mémoire utilisée par la liste

**int add\_left(Liste \*queue, Liste \*tokens);**  
Déplace le prochain token dans la queue en tête de la liste de tokens

**int add\_right(Liste \*queue, Liste \*tokens);**  
Déplace le prochain token dans la queue en fin de la liste de tokens

**int check\_combinations(Liste \*lst);**  
Cherche les chaines de plus de 3 tokens consécutifs d'une même couleur ou forme, les retire et renvoie le score associé aux différentes chaines retirées

**void shift\_commonshape\_left(Liste \*lst, Token \*tok);**  
Effectue un décalage circulaire vers la gauche des tokens de la même forme

**void shift\_commoncolor\_left(Liste \*lst, Token \*tok);**  
Effectue un décalage circulaire vers la gauche des tokens de la même couleur

## graphique.c

**Case mouse\_to\_square(int mousex, int mousey);**  
Transforme le pixel d'un clic de souris en coordonnées de case

**void window\_open(int larg, int haut);**  
Ouvre une fenêtre vide à fond noir

**void token\_draw(Token t, Case c);**  
Dessine un token de forme et couleur donnés

**void token\_draw\_list(Liste lst\_tokens, int nb\_tokens, Case cible);**  
Dessine une liste de tokens sur une ligne

**void button\_add\_draw(Case cible, char dir) ;**  
Dessine un bouton avec une flèche dans la direction voulue

**int button\_add\_check(int sizex, int sizey, Case cible);**  
Vérifie si un bouton d'ajout de jeton (gauche ou droit) a été cliqué

**int token\_select\_check(int sizex, int sizey, int nb\_tokens, Case cible, Liste lst\_tokens);**  
Vérifie si un token est cliqué

**void token\_select\_draw(Case cible, Liste lst\_tokens, int nb\_tokens, int position);**  
Si la position donnée est strictement positive, dessine la sélection

**void score\_cat(char\* message, int score);**  
Concatène le score à la fin du message

**void timer\_cat(char\* message, int timer);**  
Concatène le timer à la fin du message

**float time\_usec(struct timeval debut);**  
Renvoie le float de l'interval écoulé entre debut et maintenant

**void refresh\_screen(int sizex, int sizey, Game g, Case cible);**  
Rafraichit l'affichage de base du jeu

**void clock\_draw(Case cible, float duree);**  
Dessine une horloge pour une certaine durée

## threetogo.c

**int game\_init(Game \*game);**Initialise une partie

**void game\_free(Game \*game);**Libère tout l'espace mémoire utilisé par la partie

**int game\_loop(Game \*game);**Contient la boucle principale du jeu

# Fonctionnement d’une boucle

## Rafraichissement de l’affichage

## Gestion et interprétation des événements

## Mise à jour des informations de la partie

# Limites du programme

## Fonctionnalités implémentées

## Problèmes non-pris en charge

# Pistes d’amélioration