

Première partie

Affichage

Chapitre 1

Menu.c

```
void mainMenu(){
    /*
        Afficher menu principal
        1 - Nouvelle Partie
        2 - Charger une Partie
        3 - Quitter
    */
}

void gameMenu(int listUnitP, movable, attackable){
    /*
        Afficher le menu de jeu
        1 - Unite pouvant se deplacer -> unitMenu();
        2 - Unite pouvant attaquer -> unitMenu();
        3 - Changer de direction -> unitMenu();
        4 - Passer tour
        5 - Abandonner la partie
    */
}

void unitMenu(int choice, int listUnitP, movable, attackable){
    /*
        Afficher la liste des unites pouvant faire quelque chose
        Si choice-> 1 alors liste des unites pouvant se deplacer
        Si choice -> 2 alors liste des unites pouvant attaquer
        Si choice -> 3 alors liste de toutes les unites du joueur
        Appui sur une touche fait appel a playTurn();
    */
}
```

Chapitre 2

Grid.c

```
void gridDisp(){  
    // Affiche la grille  
}
```

Deuxième partie

Moteur de jeu

Chapitre 3

gameEngine.c

```
int gameInit(int listUnitP, int * noPlayer){
    gridInit(); // Retourne un code d'erreur
    playerInit(listUnitP); // Retourne un code d'erreur
    * noPlayer = rand(1,2);
    return true; // Si les fonctions ne retournent rien
}

vector selectUnit(int unitSelected, int listUnitP){
    // Selectionne l'unité -> coordonnees dans la matrice sous
    // forme de vecteur
}

void playTurn(int unitSelected, int listUnitP, movable){
    selectUnit(int unitSelected, listUnitP); // Renvoie les
    // coordonnees de l'unité selectionnee
    playMove(unitSelected);
    playAttack(unitSelected, listUnitP);
    playDirection(unitSelected, listUnitP);
}

bool endGame(int listUnitP){
    /*
    Test les conditions de victoire ou de match nul
    Affiche le vainqueur le cas echeant
    */
}

int movable(unitAction movable[]){
    /*
    Retourne nombre d'unités déplaçable + liste maj des unités
    pouvant se déplacer
    */
}

void timeTurn(){
    // Gestion du temps
}

int lookAround(vector currentUnit){
    /*
    Regarde autour de l'unité selectionnee si elle peut se
    déplacer sur une case si TP pas permise alors
    déplacement impossible
    */
}
```

```
void playMove(vector unitSelected, path[]){  
    /*  
        Deplace l'unité selectionnee a l'endroit desire en prenant  
        en compte les chemins possibles  
    */  
}  
  
void playAttack(vector unitSelected, int listUnitP){  
    /*  
        Attaque avec l'aide de l'unité selectionnee une unité  
        contenu dans la liste listUnitP  
    */  
}  
  
void playDirection(vector unitSelected){  
    //Change la direction de l'unité selectionnee  
}
```