Projet Sudoku – IN 301 Langage C

Table des matières

Le but de ce projet est de faire une interface permettant de jouer au sudoku.

1 Rappel de ce qu'est un sudoku

Un sudoku est une grille carrée 9×9 où chaque case peut contenir les chiffres de **1** à **9**. Au départ certaines cases contiennent un chiffre, les autres sont vides. Le but est de mettre un chiffre dans chacune des cases vides en respectant les contraintes suivantes :

- Pas deux fois le même chiffre dans une même ligne.
- Pas deux fois le même chiffre dans une même colonne.
- Pas deux fois le même chiffre dans une même région. Le sudoku est divisé en neufs régions qui forment chacune une grille 3×3.

Plus d'informations sur le sudoku : `https://fr.wikipedia.org/wiki/Sudoku`.

On appelle **case de départ** une case dans laquelle il y a un chiffre au départ et qui n'est pas modifiable et on appelle **case de travail** une case dans laquelle il n'y a pas de chiffre au départ et qui est modifiable.

2 Interface graphique

Pour l'interface vous devrez utiliser la librairie `uvsqgraphics`. Le joueur aura deux actions possibles : cliquer dans une case ou appuyer sur une des touches **U**, **V**, **S** ou **Q**.

2.1 Clic dans une case

Un clic dans une case de départ ne fait rien.

Un clic dans une case de travail modifie le contenu de la case en suivant la séquence :

Vide → **1** → **2** → **3** … **8** → **9** → **Vide**

Si dans une case de travail une valeur n'est pas possible (cette valeur est déjà présente dans la même ligne, la même colonne ou la même région), alors cette valeur n'est pas dans la séquence. Par exemple si dans une case, seules les valeurs **2** et **7** sont possibles, la séquence de clics donnera :

Vide → **2** → **7** → **Vide**

2.2 Appui sur U

L'appui sur cette touche fait un **u**ndo, c'est à dire annule la dernière action de changement de valeur d'une case. Le joueur pourra alors continuer à partir de cet état. Des appuis successifs sur **U** annulent les dernières actions en ordre inverse de l'ordre dans lequel elles ont été effectuées.

2.3 Appui sur V

L'appui sur cette touche trou**v**e une solution pour remplir l'ensemble du sudoku en respectant les règles et affiche cette solution. Si aucune solution n'existe vous devrez afficher un message le signifiant Vous devrez expliquer dans le compte-rendu le principe de résolution que vous utilisez.

2.4 Appui sur S

L'appui sur cette touche **s** auve l'état du sudoku dans un fichier. Voir la section **??** pour une description du format et du nom de fichier.

2.5 Appui sur Q

L'appui sur cette touche **q** uitte le programme.

2.6 Affichage

Vous devrez afficher en haut le nom du fichier et en dessous la grille du sudoku et rien d'autre.

Il faudra afficher la grille avec des traits plus gros pour séparer les régions.

Il faudra afficher de couleurs différentes les valeurs qui sont dans les cases de départ et celles qui sont dans les cases de travail.

Si dans une case aucune valeur n'est possible il faudra le signaler par un affichage spécifique, par exemple en mettant le fond de la case en rouge.

3 Fichiers et format de fichiers

Le format de base pour stocker les fichiers de sudoku est un fichier texte comme le fichier ci-dessous :

`exemple.sudoku`

```
..7...2..
.156.9.4.
24.1.....
5.24.1.3.
16..8..52
.9.3.21.6
.....4.83
.2.7.851.
..9...7..
```

Les `.` désignent les cases qui sont vides et les valeurs donnent les valeurs des cases de départ. Il y a donc neuf lignes qui contiennent chacune dix caractères, en comptant le retour à la ligne.

La sauvegarde en cours de jeu avec la touche **S**, se fera dans un fichier qui s'appelle `exemple.001.sudoku`, si le fichier de départ s'appelle `exemple.sudoku`.

Si le fichier de départ s'appelle `exemple.xxx.sudoku` où `xxx` est un nombre entre `001` et `999`, la sauvegarde en cours de jeu avec la touche **S**, se fera dans un fichier qui s'appelle `exemple.yyy.sudoku`, avec `yyy` qui vaut le nombre `xxx` augmenté de 1.

Dans les fichiers `exemple.xxx.sudoku` il faut stocker les valeurs des cases de départ et celles des cases de travail. Pour une case de travail qui a une valeur, par exemple 3 on écrira `*3`.

Dans le fichier `exemple.001.sudoku` ci-dessous, deux cases de travail ont une valeur, la deuxième case de la première ligne qui vaut **3** et la cinquième case de la septième ligne qui vaut **9**.

`exemple.001.sudoku`

```
.*37...2..
.156.9.4.
24.1.....
5.24.1.3.
```

```
16..8..52
.9.3.21.6
....*94.83
.2.7.851.
..9...7..
```

Dans ce format, il y a neuf lignes, mais le nombre de caractères par ligne varie puisque chaque case de travail qui a une valeur occupe deux caractères (l'étoile et la valeur).

4 Ligne de commande

La ligne de commande sera de la forme :
`./sudoku nom.sudoku`
ou bien `./sudoku nom.002.sudoku`.
Le mot `nom` pouvant être ce que l'on veut.

5 Documents fournis et travail à faire

Il vous est fourni un dossier `NOM_Prenom` donnant découpage en fichiers. Il est conseillé de respecter ce découpage ou de vous en inspirer fortement. Il est également fourni un Makefile cohérent avec le découpage.

Les différents fichiers sont :

- `gestion_sudoku.h` contient la définition du type `SUDOKU` et les signatures des fonctions de manipulation du type `SUDOKU`.
- `gestion_sudoku.c` contient le code des fonctions de manipulation du type `SUDOKU`.
- `afficher.c`, `afficher.h` contiennent les fonctions permettant l'affichage du sudoku.
- `lire_ecrire.c`, `lire_ecrire.h` contiennent les fonctions permettant la lecture depuis un fichier de sudoku et la sauvegarde d'un sudoku dans un fichier.
- `sudoku.c` contient la fonction `main`.

Il est conseillé de réaliser le projet dans l'ordre suivant :

5.1 ÉTAPE 1 – Lecture et affichage

1. définir le type sudoku ;
2. écrire la fonction de lecture d'un fichier contenant un sudoku avec uniquement des cases de départ ;
3. écrire la ou les fonctions d'affichage d'un sudoku.

À ce stade il est possible d'avoir une fonction `main` du genre :

```
int main (int argc, char *argv[]) {
    SUDOKU S;
    S = lire_fichier(argv[1]);
    initialiser_fenetre_graphique();
    sudoku_afficher(S);
    terminer_fenetre_graphique();
    exit(0);
    }
```

5.2 Étape 2 – Jouer

4. écrire une fonction qui prend en argument le sudoku et les coordonnées d'une case et modifie le contenu de cette case si c'est une case de travail en lui mettant la prochaine valeur disponible. Si c'est une case de départ, cette fonction ne fera rien ;

5. écrire la fonction `jouer` qui attend une entrée un clic et qui calcule les coordonnées de la case cliquée et qui modifie cette case.

Évidemment cela nécessite l'écriture de tout un tas d'autres fonctions annexes. À ce stade il est possible d'avoir dans le fichier `sudoku.c` quelque chose du genre :

```
SUDOKU jouer(SUDOKU S) {
POINT P = wait_clic();
int ligne = ; // À écrire
int colonne = ; // À écrire
S = sudoku_modifier_case(S,i,j);
return S;
}


int main (int argc, char *argv[]) {
    SUDOKU S;
    S = lire_fichier(argv[1]);
    initialiser_fenetre_graphique();
    sudoku_afficher(S);
    while (1) {
        S = jouer(S);
        sudoku_afficher(S);
    }
    terminer_fenetre_graphique();
    exit(0);
    }
```

5.3 Étape 3 – Sauvegarde et fin du jeu

6. écrire une fonction qui sauvegarde l'état du jeu dans un fichier, modifier la fonction de lecture pour lire les fichiers avec des cases de travail non vide ;
7. écrire une fonction qui vérifie si on a terminé le sudoku ;
8. écrire une fonction qui affiche "GAGNÉ" quand on a fini le sudoku ;
9. écrire une fonction qui quitte le programme ;
10. modifier la fonction `jouer` et la fonction `main` pour ajouter ces fonctionalités.

5.4 Étape 4 – Undo

11. écrire la gestion du undo.

5.5 Étape 5 – Résolution

12. écrire la résolution du sudoku. Vous afficherez les valeurs trouvées par la résolution d'une couleur différente. La résolution doit partir de l'état actuel du sudoku, il se peut donc qu'il n'y ait pas de solution si l'utilisateur a déjà mis des valeurs qui ne sont pas compatibles avec une solution.

6 Dates de remise et de soutenance et documents à rendre

Le projet est à faire seul.

Le projet est à remettre sur e-campus avant le dimanche **06 janvier 2019 23h59**. Une soutenance sera organisée dans la semaine du **lundi 07 au vendredi 11 janvier 2019**.

Vous devez rendre un zip contenant :

— un compte-rendu en pdf de 2 à 5 pages expliquant ce que vous avez fait. Ce pdf devra s'appeler `NOM_Prénom.pdf`

— Le dossier `NOM_Prenom` contenant votre code, en mettant évidemment votre NOM et votre Prénom.

A Exemple d'affichage

FIGURE 1 – Un exemple d'affichage. En plus de ce qui est demandé, il est affiché en bas de chaque case la liste des valeurs encore possibles pour chaque case.