

SAE 1.01 - Décomposition algorithmique

Tom Moriceau - 1C2

Liste descriptive des procédures et fonctions du Sudoku :

procédure nouvelleGrille

sélectionne une grille de sudoku et insère ses valeurs de départ dans le tableau passés en paramètres.

paramètres

lignes (Entrée) : entier, nombre de lignes (= de colonnes) de la grille

tabgrille (Sortie) : tableau, tableau qui contiendra les valeurs de la grille

procédure afficherGrille

affiche la grille du Sudoku en console grâce au tableau passés en paramètres

paramètres

lignes (Entrée) : entier, nombre de lignes (= de colonnes) de la grille

tabgrille (Entrée) : tableau, tableau contenant les valeurs à afficher dans la grille

procédure nettoyerConsole

efface les dernières lignes de la console

paramètres

pas de paramètres

fonction ajouterValeur

ajout d'une valeur dans le tableau (la grille)

paramètres

lignes (Entrée) : entier, nombre de lignes (= de colonnes) de la grille

tabgrille (Sortie) : tableau, tableau contenant les valeurs de la grille

colonne (Entrée) : colonne choisie par l'utilisateur

nombre (Entrée) : nombre choisi par l'utilisateur

ligne (Entrée) : ligne choisie par l'utilisateur

résultat

booléen, vrai si la valeur a pu être ajoutée sinon faux

fonction *verifierFin*

vérifie si la partie est terminée

paramètres

lignes (Entrée) : entier, nombre de lignes (= de colonnes) de la grille

tabgrille (Entrée) : tableau, tableau qui contient les valeurs à vérifier (les valeurs de la grille)

résultat

booléen, vrai si la partie est terminée sinon faux

Pseudo-code de l'algorithme principal :

(vous trouverez une version en noir et blanc plus loin dans le document, si jamais ce format ne vous plait pas)

```
programme sudoku c'est

// Les procédures et fonctions se retrouveront ici, voir les fiches de
// spécification ci-dessus

type t_grille := tableau[NB_LIGNES][NB_LIGNES] de entier;

constante entier NB_LIGNES := 9 // Nombre de lignes (et donc de colonnes)

début

    grille_actuelle : t_grille;
    partie_terminee : booléen;
    choixcol, choixnb, choixligne : entier;
    rejouer : caractère;

    partie_terminee := faux;

    nouvelleGrille(entE NB_LIGNES, sortE grille_actuelle)
    écrireEcran("Bienvenue dans le Sudoku ! Voici votre grille : \n\n");
    afficherGrille(entE NB_LIGNES, entE grille_actuelle)
```

```

// Deroulement de la partie
tant que (partie_terminee == faux) faire

    // Tant que le choix de l'utilisateur n'est pas correct compte tenu
    des règles du jeu
    écrireEcran("Dans quelle colonne voulez-vous placer votre chiffre ?
[1-9] : ");
    lireClavier(choixcol);
    tant que (choixcol < 1 ou choixcol > NB_LIGNES) faire
        écrireEcran("Erreur : La valeur entrée doit être un entier compris
entre 1 et 9. Veuillez réessayer.");
        écrireEcran("Dans quelle colonne voulez-vous placer votre chiffre
? [1-9] : ");
        lireClavier(choixcol);

    finfaire

    lettreLigne : caractère; // lettre correspondant à la ligne choisie
    par l'utilisateur
    choixligne := 0
    faire
        écrireEcran("Dans quelle ligne voulez-vous placer votre chiffre ?
[A-I] : ");
        lireClavier(lettreLigne);
        selon lettreLigne c'est
            quand 'A' => choixligne = 1;
            quand 'B' => choixligne = 2;
            quand 'C' => choixligne = 3;
            quand 'D' => choixligne = 4;
            quand 'E' => choixligne = 5;
            quand 'F' => choixligne = 6;
            quand 'G' => choixligne = 7;
            quand 'H' => choixligne = 8;
            quand 'I' => choixligne = 9;
            quand autre => écrireEcran("Erreur : La valeur entrée doit être
comprise entre A et I dans l'alphabet. Attention aux majuscules. Veuillez
réessayer.\nDans quelle ligne voulez-vous placer votre chiffre ? [A-I]
:");
        finselon
    tant que(choixligne == 0);

```

```

        écrireEcran("Quel chiffre voulez-vous placer ? [1-9] : ");
        lireClavier(choixnb);
        tant que (choixnb < 1 ou choixnb > 9) faire
            écrireEcran("Erreur : La valeur entrée doit être un entier compris
entre 1 et 9. Veuillez réessayer.");
            écrireEcran("Quel chiffre voulez-vous placer ? [1-9] : ");
            lireClavier(choixcol);
        finfaire

        // Ajout de la valeur dans la grille en passant par les verifications
des regles du jeu
        tant que(ajouterValeur(entE NB_LIGNES, sortE grille_actuelle, entE
choixcol, entE choixnb, entE choixligne) == faux)

            écrireEcran("Chiffre placé.")
            afficherGrille(entE NB_LIGNES, entE grille_actuelle)

        // Condition de victoire
        si(verifierFin(entE, NB_LIGNES, entE grille_actuelle) == vrai):
            écrireEcran("Bravo, vous avez réussi à compléter ce Sudoku !
Souhaitez-vous rejouer ? [O/N] :");
            lireClavier(rejouer);

            si(rejouer == "O")
                nouvelleGrille(entE NB_LIGNES, sortE grille_actuelle) // création
d'une nouvelle grille différente
                nettoyerConsole()
                // Nouvelle iteration de la boucle tant que, avec une nouvelle
grille de sudoku

            sinon si(rejouer == "N")
                partie_terminee := vrai
                écrireEcran("Très bien, merci d'avoir joué !");
            finsi
        finsi

    finfaire

fin

```

Version en noir et blanc :

programme sudoku c'est

// Les procédures et fonctions se retrouveront ici, voir les fiches de spécification ci-dessus

type t_grille := tableau[NB_LIGNES][NB_LIGNES] de entier;

constante entier NB_LIGNES := 9 // Nombre de lignes (et donc de colonnes)

début

grille_actuelle : t_grille;
partie_terminee : booléen;
choixcol, choixnb, choixligne : entier;
rejouer : caractère;

partie_terminee := faux;

nouvelleGrille(entE NB_LIGNES, sortE grille_actuelle)
écrireEcran("Bienvenue dans le Sudoku ! Voici votre grille : \n\n");
afficherGrille(entE NB_LIGNES, entE grille_actuelle)

// Deroulement de la partie

tant que (partie_terminee == faux) faire

 // Tant que le choix de l'utilisateur n'est pas correct compte tenu des règles du jeu
 écrireEcran("Dans quelle colonne voulez-vous placer votre chiffre ? [1-9] : ");
 lireClavier(choixcol);
 tant que (choixcol < 1 ou choixcol > NB_LIGNES) faire
 écrireEcran("Erreur : La valeur entrée doit être un entier compris entre 1 et 9. Veuillez
réessayer.");
 écrireEcran("Dans quelle colonne voulez-vous placer votre chiffre ? [1-9] : ");
 lireClavier(choixcol);

finfaire

lettreLigne : caractère; // lettre correspondant à la ligne choisie par l'utilisateur
choixligne := 0

faire

 écrireEcran("Dans quelle ligne voulez-vous placer votre chiffre ? [A-I] : ");
 lireClavier(lettreLigne);

```

selon lettreLigne c'est
    quand 'A' => choixligne = 1;
    quand 'B' => choixligne = 2;
    quand 'C' => choixligne = 3;
    quand 'D' => choixligne = 4;
    quand 'E' => choixligne = 5;
    quand 'F' => choixligne = 6;
    quand 'G' => choixligne = 7;
    quand 'H' => choixligne = 8;
    quand 'I' => choixligne = 9;
    quand autre => écrireEcran("Erreur : La valeur entrée doit être comprise entre A et I dans
l'alphabet. Attention aux majuscules. Veuillez réessayer.\nDans quelle ligne voulez-vous placer
votre chiffre ? [A-I] :");
    fin selon
    tant que(choixligne == 0);

    écrireEcran("Quel chiffre voulez-vous placer ? [1-9] : ");
    lireClavier(choixnb);
    tant que (choixnb < 1 ou choixnb > 9) faire
        écrireEcran("Erreur : La valeur entrée doit être un entier compris entre 1 et 9. Veuillez
réessayer.");
        écrireEcran("Quel chiffre voulez-vous placer ? [1-9] : ");
        lireClavier(choixcol);
    fin faire

// Ajout de la valeur dans la grille en passant par les verifications des regles du jeu
    tant que(ajouterValeur(entE NB_LIGNES, sortE grille_actuelle, entE choixcol, entE choixnb,
entE choixligne) == faux)

    écrireEcran("Chiffre placé.")
    afficherGrille(entE NB_LIGNES, entE grille_actuelle)

// Condition de victoire
    si(verifierFin(entE, NB_LIGNES, entE grille_actuelle) == vrai):
        écrireEcran("Bravo, vous avez réussi à compléter ce Sudoku ! Souhaitez-vous rejouer ?
[O/N] :");
        lireClavier(rejouer);

    si(rejouer == "O")
        nouvelleGrille(entE NB_LIGNES, sortE grille_actuelle) // création d'une nouvelle grille
différente
        nettoyerConsole()
        // Nouvelle iteration de la boucle tant que, avec une nouvelle grille de sudoku

```

```
    sinon si(rejouer == "N")
        partie_terminee := vrai
        écrireEcran("Très bien, merci d'avoir joué !");
    finsi
finsi

finfaire

fin
```