Université des Antilles et de la Guyane
Département Scientifique Interfacultaire
Corrigé Examen INFO1
Informatique

Le corrigé propose **une** des réponses possibles aux problèmes posés. Le but de l'examen n'étant pas de traiter le problème d'une manière précise (sauf si indications dans ce sens), les solutions différentes mais justes auront également été retenues comme réponses correctes. En particulier, les algorithmes demandés pour le jeu de la vie pouvaient être écrit en langage 'C' ou algorithmique.

Somme des entiers pairs

Exercice 1*

Écrire une fonction qui retourne la somme des entiers pairs inférieurs ou égaux à un entier n donné.

```
int sommePairs(int n) {
      int i, somme = 0;
      for(i=0 ; i<=n ; i++) {
            if (i % 2 == 0) somme += i ;
      }
      return somme;
}
```

Le jeu de la vie.

NB : Chaque exercice peut être traité indépendamment.

On dispose d'un échiquier, dans lequel les cases peuvent prendre les valeurs 0 ou 1.
À chaque étape, on calcule la nouvelle valeur de chacune des cases en fonction des valeurs des cases voisines. Les cases voisines d'une case étant celles se trouvant de part et d'autre de cette case sur la même ligne ou sur la même colonne *(pas en diagonale)*.

La valeur d'une case passe à 0 si elle possède 0 ou 4 voisins à 1 (elle meurt isolée ou étouffée).
La valeur d'une case passe à 1 si elle possède 2 ou 3 voisins à 1.
Elle ne change pas dans les autres cas.

Pour éviter les cas particuliers des cases se trouvant en périphérie de l'échiquier, on considérera que ce dernier est stocké dans un tableau comprenant un bordure remplie de zéros (0) :

0	*0*	*0*	*0*	*0*	*0*
0	**1**	**0**	**1**	**0**	*0*
0	**1**	**1**	**0**	**1**	*0*
0	**1**	**1**	**0**	**0**	*0*
0	**1**	**0**	**0**	**1**	*0*
0	*0*	*0*	*0*	*0*	*0*

Dans l'échiquier représenté ici (*avec sa bordure)*, la case à l'intersection de la 2ème ligne et de la 3ème colonne à 3 voisins à 1.

Le programme principal vous est donné et est le suivant :

```
void main()
{
    int n = 25;
    int M[n][n];     /* on déclare un tableau carré */
    int i,NbEtapes;

    configInit(M);
    printf("Entrer le nombre d'etapes : ");
    scanf("%d",&NbEtapes);
    for (i=1;i<=NbEtapes;i++)
    {
      changeEtat(M);
      imprime(M);
    }
}
```

Exercice 2*

On souhaite pouvoir visualiser l'évolution du jeu. Écrivez la procédure `imprime` qui prend en paramètre un tableau stockant un échiquier et affiche ce dernier à l'écran.

```
void imprime(int M[n][n])
{
    int i,j;

    for (i=1;i<n-1;i++)
    {
      printf("\n");
      for (j=1;j<n-1;j++) printf("%1d ",M[i][j]);
    }
    printf("\n");
}
```

Exercice 3*

On souhaite pouvoir calculer pour une case de l'échiquier le nombre de voisins à 1 afin de tester les différents cas. Écrivez l'algorithme permettant d'obtenir ce nombre.

```
int nbVoisins(int M[n][n], int ligne, int colonne) {
    return(M[i-1][j]+M[i+1][j]+M[i][j-1]+M[i][j+1]);
}
```

Exercice 4**

On dispose d'une fonction binRandom qui retourne au hasard la valeur 0 ou la valeur 1. On souhaite initialiser **l'échiquier** avec une valeur au hasard pour chacune des cases. Écrivez la procédure `configInit` qui permet d'initialiser **le tableau**, c'est-à-dire non seulement initialiser l'échiquier mais aussi la bordure.

```
void configInit(int M[n][n])
{
    int i,j;

    /* Mise a 0 des contours */
    for (i=0;i<n;i++)
    {
```

```
      M[0][i]=0; M[n-1][i]=0;
      M[i][0]=0; M[i][n-1]=0;
    }
    /* Acquisition */
    //printf("Entrer la configuration initiale :\n");
    for (i=1;i<n-1;i++)
      for (j=1;j<n-1;j++)
          M[i][j] = rand() > (RAND_MAX/2.0) ? 1 : 0 ;
          //scanf("%d",&M[i][j]);
}
```

Exercice 5**

En utilisant les fonctions et procédures précédentes, écrivez la fonction `estVivante` qui retourne la valeur que devra prendre une case à l'étape suivante.

```
int estVivante(int M[n][n], int ligne, int colonne) {
    NbVois = nbVoisins(M,ligne,colonne);
          if ((NbVois==4) || (NbVois==0))
            return 0;
          else
            if (NbVois==2)
                return 1;
            else return M[ligne][colonne];
}
```

Exercice 6***

Complétez la procédure suivante qui permet de faire évoluer l'ensemble de l'échiquier d'une étape vers une autre.

```
void changeEtat(int M[n][n])
{
    int i,j,NbVois;
    int Maux[n][n];

    for (i=1;i<n-1;i++)
      for (j=1;j<n-1;j++)
      {
          Maux[i][j] = estVivante(M, i, j);
      }
    /* Recopie de MAux dans M */
    for (i=1;i<n-1;i++)
      for (j=1;j<n-1;j++)
            M[i][j]=MAux[i][j];
}
```

Programmation en C

Exercice 7*

Dans le programmes C suivant, donnez les erreurs et les corrections à effectuer.*

```
int addition(int a, int b)
{
    int a, b, c;
    c = a + b ;
    return c;
}
```

Exercice 8**

Que réalise les programmes suivant : **

1)
```
for (i=0;i<10;i++);
    printf("hello world !\n");
```

Boucle 10 fois puis affiche « hello world !» à l'écran.

2)
```
for (i=0,i<10,i++)
    printf("hello world !\n");
```

Erreur à la compilation à cause des virgules dans l'instruction for.

Exercice 9***

Que réalise le programme C suivant (répondez aux questions suivantes). ***

- Que fait la fonction `t` ?

La fonction t compare deux entiers, retourne 0 si le premier est inférieur au second, retranche le second du premier et retourne 1 sinon.

- La variable `cont` est déclarée en C comme un `int`, mais quel est *réellement* son type.

La variable *cont* est en fait une variable booléenne puisqu'elle est utilisée en tant que telle dans l'instruction *for*.

- Quel relation lie les variables `i`, `a`, `b` et `c` à chaque passage dans la boucle `for`.

La relation qui relie i, a, b et c est c = b*(i+1) + a.

- Completez l'instruction `printf` en fin du programme principal (main).

```
printf("La quotient de la division de %d par %d est %d et le reste est %d\n"
, b*i+a, b, i, a);
```

```
int i, a, b, cont = 1;

int t(int *a, int b){
    if (*a < b) return 0;
    *a = *a - b;
    return 1
}

main(){
```

```
    int i, a, b, c, cont = 1;

    printf("Entrez une valeur pour a\n");
    scanf("%d", &a);
    c = a;
    printf("Entrez une valeur pour b\n");
    scanf("%d", &b);

    for (i = -1 ; cont ; i++) cont = t(&a, b);

    printf("...\n", ..., ..., ..., ...);
}
```

Parcours en accordéon

Exercice 10****

Écrire une fonction qui ***affiche*** un parcours en accordéon d'un tableau à deux dimensions. Plus précisément, on parcourt la première ligne de gauche à droite puis la seconde de droite à gauche et ainsi de suite en alternant le sens de parcours des lignes.

Exemple :

Pour le tableau suivant,

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

On doit obtenir l'affichage suivant :

0, 1, 2, 3, 7, 6, 5, 4, 8, 9, 10, 11, 15, 14, 13, 12, 16, 17, 18, 19

```
void accordeon(tab[m][n] tableau, int m, int n) {
    for (i=0 ; i<m ; i++) {
      if (i%2 == 0) {
          debut = 0 ;
          pas   = 1 ;
          fin   = n ;
        } else {
          debut = n-1 ;
          pas   = -1 ;
          fin   = -1 ;
        }
      while (debut != fin) {
            printf("\t%i", tableau[i][j]);
          debut += pas;
      }
      printf("\n");
   }
}
```
