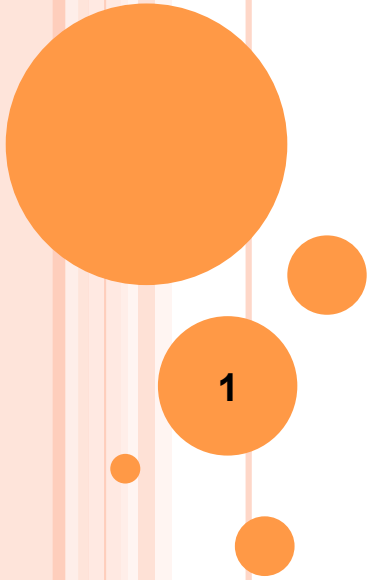
Introduction au projet
Le jeu de la vie



Sujet conçu par Florence Zara

LE JEU DE LA VIE : UN AUTOMATE CELLULAIRE

- Un automate cellulaire est une grille de cellules (un tableau à 2 dimensions, les cellules sont les cases du tableau).
- Chaque cellule a un état qui évolue au cours du temps.
- L'état d'une cellule au temps *t+1* est fonction de l'état au temps *t* d'un nombre fini de cellules appelé son « voisinage ».
- À chaque nouvelle unité de temps, les mêmes règles sont appliquées simultanément à toutes les cellules de la grille, produisant une nouvelle « génération » de cellules dépendant entièrement de la génération précédente.

Règles du jeu de la vie

1. Une cellule morte possédant exactement trois voisines vivantes devient vivante (elle naît).
2. Une cellule vivante possédant deux ou trois voisines vivantes le reste, sinon elle meurt.

Simulation

COMMENT REPRÉSENTER UN TABLEAU À 2 DIMENSIONS ?

- On représentera un tableau à 2 dimensions par une liste (de longueur NbLignes) de listes (de longueur NbColonnes)
- Par analogie avec une double-boucle (LIF1), pour le parcourir nous devons utiliser 2 niveaux de récursivité :
  - l'un sur les colonnes
  - l'autre sur les lignes
- Exemple :
  - AfficheUneLigne pour afficher une ligne du tableau (récursivité sur les colonnes)
  - AfficheTableau pour afficher l'ensemble du tableau (récursivité sur les lignes et appel à AfficheUneLigne)

CE QUE VOUS ALLEZ RÉALISER

```
> (Play 3 jeu-tableau)
------ Jeu de la vie ------
Pas de temps courant : 1
Etat actuel du jeu de la vie :
*+++++
**+++*+
+**+*+
++***+


Voisinage des cellules :
231111
343312
333533
123322


Changement d etat des cellules :
*B++++
*DBBD+
B**+*B
++B**+
```

```
----------
Pas de temps courant : 2
Etat actuel du jeu de la vie :
**++++
*+**++
***+**
++***+


Voisinage des cellules :
233210
474332
255742
243433


Changement d etat des cellules :
**B+++
D+D*B+
*DD+D*
++*D*B
```

```
----------
Pas de temps courant : 3
Etat actuel du jeu de la vie :
******++
++++**+
*++++*
++*+**


Voisinage des cellules :
12232122
34322222
02245322
12022222


Changement d etat des cellules :
D**B++
B+B**+
D++++*
++D+**

----------
La simulation est finie
>
```

Fonctions utiles

INTERACTIONS AVEC L'UTILISATEUR

- Afficher à l'écran : fonction prédéfinie `display`

  Ex :

```
> (display "Bonjour")
Bonjour
> (display 'Bonjour)
bonjour
> (define a 'bonjour)
> (display a)
bonjour
```

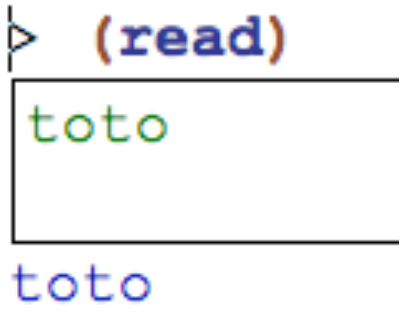
- Passer à la ligne : fonction prédéfinie `newline`

  Ex : `(newline)`

INTERACTIONS AVEC L'UTILISATEUR

- Lire au clavier : fonction prédéfinie read

Ex :



- Utilisation du résultat de la fonction :

```
(let ((reponse (read)))
        (if (eq? reponse 'o)
            'toto
            'tata))
```

SORTIR DU FONCTIONNEL

- Utilisation de la séquence : fonction prédéfinie `begin`

```
(define exemple
  (lambda ()
    (begin
      (display "Bonjour, souhaitez-vous continuer (o/n)")
      (newline)
      (let ((reponse (read)))
        (if (eq? reponse 'o)
            (exemple)
            (display "Au revoir !"))))))
```

Comment rendre le travail

COMMENT RENDRE LE TRAVAIL
VALABLE AUSSI POUR LE TP NOTÉ

- Vous envoyez par mail un fichier attaché à votre nom
  - Ex : ProjetGuin.scm TPnoteGuin.scm
- Les premières lignes de votre fichier doivent être vos nom, prénom et numéro d'étudiant (en commentaire)
- Vous ne partez pas sans vous être assuré(e) que votre enseignant a bien reçu votre fichier

COMMENT RENDRE LE TRAVAIL
VALABLE AUSSI POUR LE TP NOTÉ

- Toutes vos fonctions doivent être commentées et testées
  - Type des arguments et du résultat
  - Des noms d'arguments significatifs
  - Ce que fait la fonction (en particulier si c'est une fonction que vous introduisez)
  - Des commentaires sur certaines parties pour comprendre ce qu'elle fait
  - Elle est suivie (en commentaire) des tests effectués et des résultats obtenus
- Ceci compte pour une part importante de l'évaluation

EXEMPLE

```
; Nathalie Guin
; numéro d'étudiant : 00000007

; fonction qui calcule le nombre d'occurences
; d'un element x dans une liste l
(define mystere ; -> un entier
  (lambda (x l) ; x un element, l une liste
    (cond ((null? l) 0)
          ((eq? x (car l)) (+ 1 (mystere x (cdr l))))
                    ; on compte 1 pour l'element
          (else (mystere x (cdr l))))))

; (mystere 'a '(e a z z t)) -> 1
; (mystere 'a '(e a z a t)) -> 2
; (mystere 'a '(e z z t)) -> 0
```

POUR LE PROJET

- Il vaut mieux ne pas tout faire mais le faire bien
- Vous devez travailler chez vous en dehors des deux séances
- Vous aurez à présenter oralement votre travail à votre enseignant : on note plus votre démarche et ce que vous avez compris que ce qui marche