

# Mini Projet :

## Programmation du jeu de la vie de Conway

### Mise en situation :

Le jeu de la vie a été imaginé par le mathématicien John Conway dans les années 1970. Depuis, presque 50 ans, ce jeu fascine les mathématiciens et les informaticiens mais aussi les biologistes et les philosophes parce qu'il nous montre comment un système évoluant sous des règles simples peut engendrer des résultats incroyablement riches.

Le jeu de la vie se joue dans une grille composée de cases carrées. Si une case est coloriée en noir, on dit qu'elle contient de la vie ou une cellule vivante. Si non, elle est coloriée en blanc. L'idée du jeu de la vie est d'appliquer deux règles élémentaires simples au tour par tour et de surveiller l'évolution d'une structure de départ.

Plus d'information : <https://www.youtube.com/watch?v=S-W0NX97DB0>

### Objectif :

Réaliser en C++ un programme qui simule divers développements du jeu de la vie. Elaborer différents types de structures, suivre leurs évolutions.

### Partie I :

Mise en place de l'environnement de développement

#### 1 C++/CLI :

Commencez par installer Visual Studio 2017 qui est actuellement la dernière version disponible. Optez pour l'édition Community qui est gratuite. Installez l'environnement de développement C++ et assurez-vous de cocher C++/CLI support dans composants individuels.

C++/CLI (Common Language Infrastructure) permet de profiter du Framework .NET et notamment des composantes graphiques, ainsi on pourra créer des projets CLR (Common Language Runtime) qui pourront intégrer facilement des fenêtres (Windows Form), des zones de dessins (PictureBox), des zones de textes (TextBox), des boutons (Button), etc...

#### 2 Windows Form :

Créez un nouveau projet de type Projet Vide CLR ayant pour nom : JeuDeLaVie  
Clic droit sur le projet Propriétés -> Onglet Liens -> Système -> Sous-Système. Sélectionner Windows (/SUB...). Puis clic sur Appliquer

Clic droit sur le projet Propriétés -> Onglet Liens -> Avancé -> Point d'entrée. Entrer Main.  
Puis clic sur Appliquer  
Clic droit sur le projet Ajouter -> Onglet UI -> Windows Form  
Vous devriez avoir une erreur. Pas de soucis, fermez cette erreur et copiez le code suivant dans MyForm.cpp

```
#include "MyForm.h"

using namespace System;
using namespace System::Windows::Forms;

[STAThreadAttribute]
void Main(array<String^>^ args) {
    Application::EnableVisualStyles();
    Application::SetCompatibleTextRenderingDefault(false);
    JeuDeLaVie::MyForm form;
    Application::Run(%form);
}
```

Enregistrez, fermez et ré-ouvrez VS2017 puis click sur débbugger.  
Maintenant en double cliquant sur MyForm.h, vous pouvez ajouter des boutons, des zones de textes et des zones de dessins grâce à la Boite à outils -> Contrôles Communs.

### 3 PictureBox, TextBox, Timer et Button :

Ajoutez une pictureBox1 qui servira de zone de dessin pour notre jeu  
Ajoutez un button1 (Start) pour lancer la simulation  
Ajoutez un button2 (Stop) pour arrêter la simulation  
Ajoutez un textBox1 qui servira de point d'entrée pour choisir la structure de départ  
Ajouter un button3 (Init) pour dessiner la structure de départ  
Ajoutez un timer1 qui servira de compteur pour passer d'un tour à un autre

## Partie II :

Programmation du Jeu

### 1 Principe :

En double cliquant sur timer1, on génère la fonction timer1\_Tick qui sera le cœur de notre programme. Dans cette fonction, on mettra à jour la pictureBox1 selon les règles d'évolutions du jeu de la vie.

Le bouton Start sert à lancer timer1 : timer1->Enabled = true;

Le bouton Stop sert à stopper timer1 : timer1->Enabled = false;

Le textBox contient le nom de la structure de départ

Le bouton Init sert à dessiner cette structure de départ

```
private: System::Void button3_Click(System::Object^ sender, System::EventArgs^ e) {
    if (textBox1->Text == "escalier")
        ...// dessiner la structure escalier
}
```

Pour dessiner un carré noir à l'emplacement (0,0) et de taille (10,10) vous pouvez utiliser le code suivant :

```

pictureBox1->Image = gcnew Bitmap(700,700);
Graphics^ g;
pictureBox1->Image = gcnew Bitmap(pictureBox1->Image);
g = Graphics::FromImage(pictureBox1->Image);
g->FillRectangle(gcnew SolidBrush(Color::Black), 0, 0, 10, 10);

```

Pour un carré blanc, il suffit de changer la couleur Black en White.

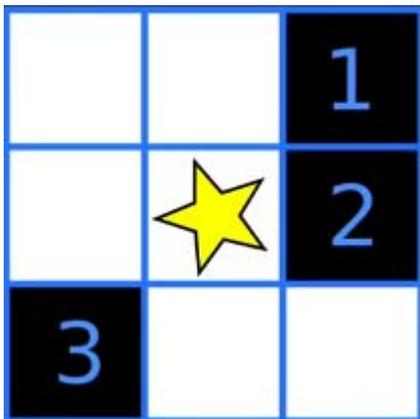
## 2 Règles du jeu :

Les règles du jeu sont :

Règle 1 : Pour les cases noires : Survie si entourée de 2 ou 3 voisines :



Règle 2 : Pour les cases blanches : Naissance si entourée de 3 voisines :



A la fin de chaque tour une mise à jour de l'ensemble des cases s'effectue en tenant compte de ces 2 règles.

## 3 Mise en place :

Avant de commencer à dessiner des structures, on se propose de dessiner le grillage qui contient l'ensemble des carrés noirs et blancs. Ajoutez dans la fonction `button3_Click`, le code qui permet de dessiner le grillage. Pour dessiner les lignes du grillage on pourra par exemple utiliser la fonction `FillRectangle` avec les bonnes proportions pour dessiner une ligne.

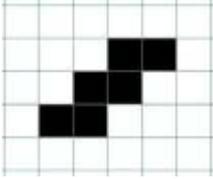
Ajoutez les conditions nécessaires dans la fonction `timer1_Tick` pour que soient satisfaites les règles du jeu de la vie. La fonction `timer1_Tick` s'exécute à chaque intervalle et sa valeur est en milliseconde. Augmentez ou diminuez la propriété `timer1->interval` de façon à accélérer ou à ralentir la vitesse de la simulation.

### Partie III :

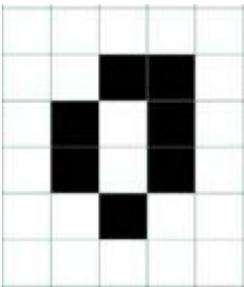
Réalisez et commentez les simulations à partir des structures de départ suivantes :

#### Structures stables :

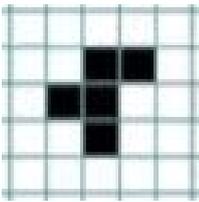
1. Escalier :



2. Ruche modifiée :

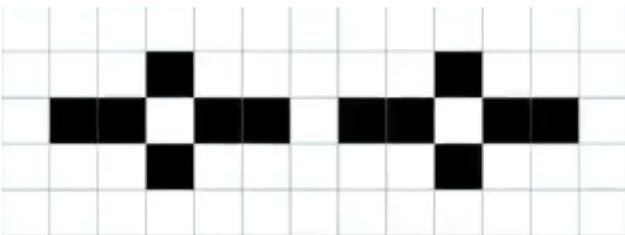


3. Structure de Conway :

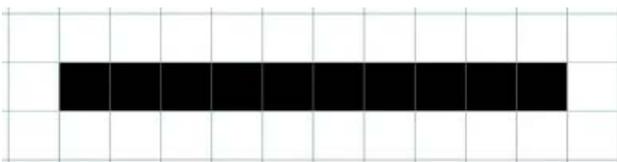


#### Oscillateurs :

4. Pulsar :

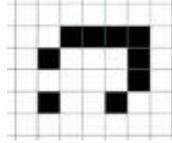


5. Penta-décathlon

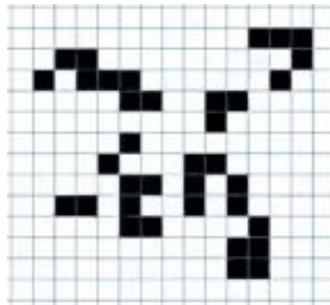


#### Vaisseaux :

6. Petit vaisseau :

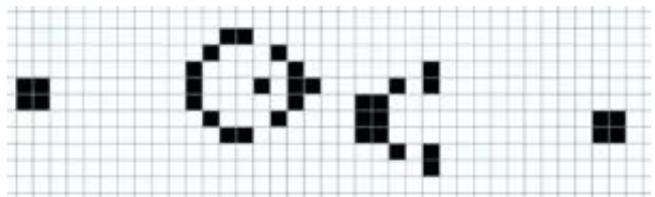


7. Oie de canada :



#### Structures infinies :

8. Canon à planeur :



9. Plus petite structure infinie :



10. Max de Tim Coe (découverte en 1995) :

