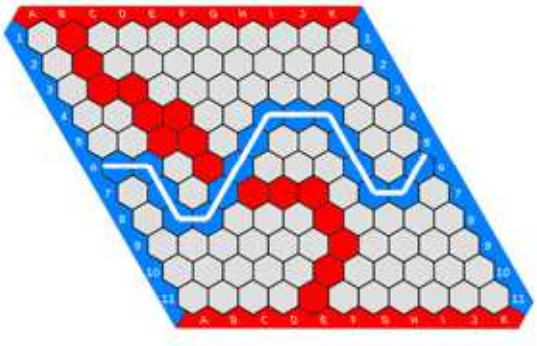
Le jeu de Hex

AMBLARD David, Ruth Tanguy



Matrice du plateau vierge :

In[59]:=

In[60]:=
```
n = 6
```
Out[60]= 6

In[61]:=
```
plateauvierge = {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}
```
Out[61]= {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}

In[62]:=
```
plateau = plateauvierge
```
Out[62]= {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}

In[63]:=
```
{{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}
```
Out[63]= {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}

In[64]:=
```
plateau =
 ReplacePart[plateau, {RandomChoice[Range[n]], RandomChoice[Range[n]]} → 1]
```
Out[64]= {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}

In[65]:=
```
s = Sum[plateau[[i, j]], {i, 1, n}, {j, 1, n}]
```

Part::partw : Part 5 of {0, 0, 0, 0} does not exist. >>

Part::partw : Part 6 of {0, 0, 0, 0} does not exist. >>

Part::partw : Part 5 of {0, 0, 0, 0} does not exist. >>

General::stop : Further output of Part::partw will be suppressed during this calculation. >>

Out[65]=
```
1 + {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦1, 5⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦1, 6⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦2, 5⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦2, 6⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦3, 5⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦3, 6⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦4, 5⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦4, 6⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦5, 1⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦5, 2⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦5, 3⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦5, 4⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦5, 5⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦5, 6⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦6, 1⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦6, 2⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦6, 3⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦6, 4⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦6, 5⟧ +
 {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 0, 0, 0}}⟦6, 6⟧
```

On a un plateau, une incrementation, et la somme des coefficients (phase d'initialisation)

```
q = {RandomChoice[Range[n]],
   RandomChoice[Range[n]]}
```

```
{4, 4}
```

```
plateau[[First[q]]][[Last[q]]]
```

```
1
```

```
plateau = ReplacePart[plateau, q → 1]
```

```
plateau
```

```
{{1, 1, 1, 1}, {1, 1, 0, 1}, {1, 1, 1, 1}, {1, 1, 1, 1}}
```

```
q = 1
```

```
1
```

```
q = {RandomChoice[Range[n]], RandomChoice[Range[n]]};
While[plateau[[First[q]]][[Last[q]]] = 1,
 q = {RandomChoice[Range[n]], RandomChoice[Range[n]]};
 Print[q]]

plateau = ReplacePart[plateau, q → 1]
```

```
{{0, 0, 0, 0}, {1, 1, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}
```

```
plateau = ReplacePart[plateau, q → 1]
```

```
{{0, 0, 0, 0}, {1, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}
```

```
Dynamic[plateau // MatrixForm]
```

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Code (mode joueur humain+random)

```
initialiser[] := (n = 4;
  plateau = Table[0, {n}, {n}];
  joueur = 1;
  gagner = False;
  Clear[pos])

initialiser[]

jouer[] :=
 (plateau = ReplacePart[plateau, RandomChoice[Position[plateau, 0]] → joueur];
  joueur = -joueur;)

jouer[{l_, c_}] :=
 If[plateau[[l, c]] == 0, (plateau = ReplacePart[plateau, {l, c} → joueur];
   joueur = -joueur;), plateau;
  Print["mauvais emplacemant"]]

jouer[{4, 3}]

jouer[]
```

Chemin

Liste de l'ensemble des positions ou apparaissent tous les 1 (un), ou -1

```
position[joueur_] := Position[plateau, joueur]
```

```
pos = position[joueur]
```

```
{{1, 1}, {1, 2}, {2, 1}, {2, 3}, {2, 4}, {3, 1}, {3, 2}, {4, 2}}
```

Voisins : cree l'ensemble des voisins existants (pour un point, en virant ceux qui contiennent un 0)

```
voisins[{c_Integer, l_Integer}] :=
 Select[{{c, l + 1}, {c, l - 1}, {c + 1, l}, {c - 1, l}, {c + 1, l - 1}, {c - 1, l + 1}},
  Not[MemberQ[#, 0]] &]
```

Fait les voisins pour une sous liste, difference par rapport a avant : on peut mettre une liste, pas rentrer les coordonnees

```
voisins[{l__List}] := Union[Flatten[Map[voisins, {l}], 1]]
```

```
voisins[{First[pos[joueur]]}]
```

```
voisins[{1}]
```

Elle fait une liste des voisins mais seulement pour une seule position

```
etend[lp_, pos_] := Union[Select[voisins[lp], MemberQ[pos, #] &], lp]
```

```
etend[etend[{First[pos]}, pos], pos]
```

```
{{1, 1}, {2, 1}}
```

Donne la liste finale pour toutes les point de departs possibles

In[77]:=
```
etendTout[pos_] := Map[First, Tally[Map[FixedPoint[etend[#, pos] &, {#}] &, pos]]]
```

In[78]:=
```
etendTout[pos]
```

```
{{{1, 2}, {1, 3}, {2, 1}, {2, 3}, {3, 1}}}
```

Verifie qui gagne

```
conditiondegagneun[pos_] :=
 If[Intersection[Map[{First[First[#]], First[Last[#]]} &, etendTout[pos]],
    {{1, n}}] == {{1, n}}, Print["joueur 1 a gagne !!"];
  gagner = True, Print["a -1 de jouer"]]
```

```
conditiondegagnemoinsun[p_] := With[{pos = Map[Reverse[#] &, p]},
  If[Intersection[Map[{First[First[#]], First[Last[#]]} &, etendTout[pos]],
     {{1, n}}] == {{1, n}}, Print["joueur -1 a gagne !!"];
   gagner = True, Print["a 1 de jouer"]]]
```

Programme

In[66]:=
```
initialiser[]; Dynamic[plateau // MatrixForm]
```

Out[66]=
$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

In[67]:=
```
Button["Joue", joue[]]
```

Out[67]= Joue

In[68]:=
```
Button["Reinitialiser", initialiser[]]
```

Out[68]= Reinitialiser

In[69]:=
```
joue[] := (pos = position[joueur];
  jouer[];
  pos = position[-joueur];
  If[joueur == 1, conditiondegagnemoinsun[pos], conditiondegagneun[pos]];)
```

In[70]:=
```
joue[{l_, c_}] := (pos = position[joueur];
  jouer[{l, c}];
  pos = position[-joueur];
  If[joueur == 1, conditiondegagnemoinsun[pos], conditiondegagneun[pos]];)
```

In[71]:=
```
joue[{3, 1}]
```

In[72]:=

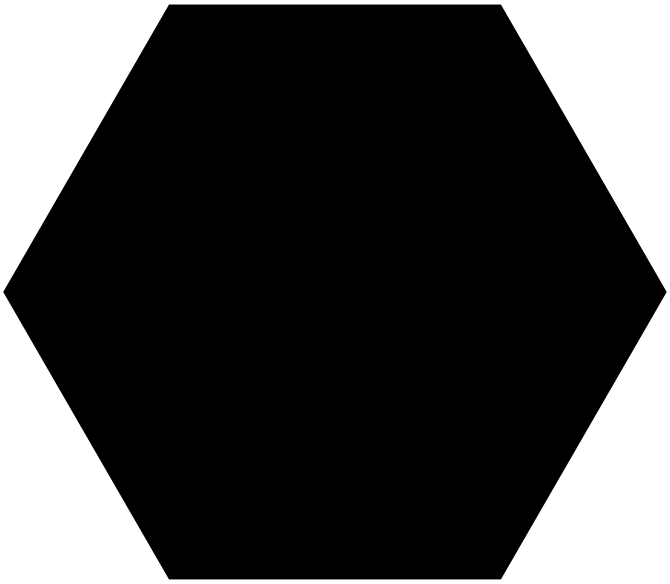
```
a = Polygon[Table[{Cos[2 π k / 6], Sin[2 π k / 6]}, {k, 0, 5}]]
```

Out[72]= $\text{Polygon}\left[\left\{\{1, 0\}, \left\{\frac{1}{2}, \frac{\sqrt{3}}{2}\right\}, \left\{-\frac{1}{2}, \frac{\sqrt{3}}{2}\right\}, \{-1, 0\}, \left\{-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right\}, \left\{\frac{1}{2}, -\frac{\sqrt{3}}{2}\right\}\right\}\right]$

In[73]:=

```
Graphics[a]
```

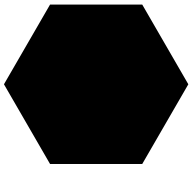
Out[73]=



In[74]:=

```
Show[Graphics[Polygon[{{1, 0}, {1/2, Sqrt[3]/2}, {-1/2, Sqrt[3]/2},
    {-1, 0}, {-1/2, -Sqrt[3]/2}, {1/2, -Sqrt[3]/2}}]], ImageSize → Tiny]
```

Out[74]=

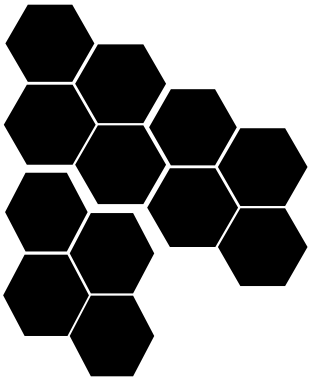


In[75]:= **Show[%1307, ImageSize → Tiny]**

Show::gtype : Out is not a type of graphics. ≫

Out[75]= Show[%1307, ImageSize → Tiny]

In[76]:=



Out[76]=