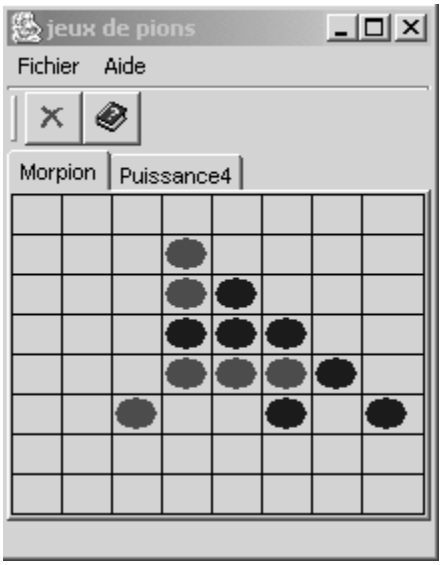
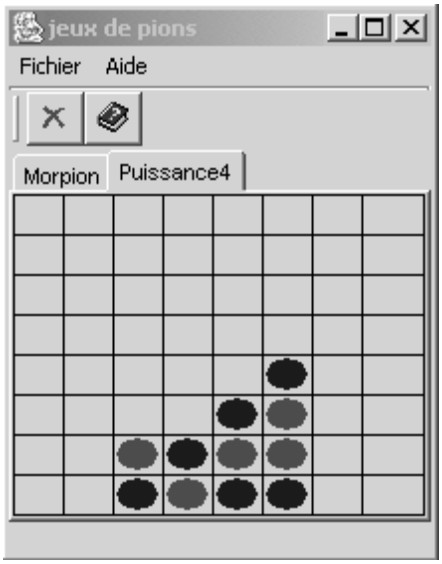
Programmation
Examen partiel

On désire réaliser des classes permettant à des joueurs de jouer à des jeux de pions. Un jeu de pions est un jeu dans lequel deux joueurs placent chacun à son tour un pion sur une grille de manière à produire des lignes de plusieurs pions : il existe plusieurs jeux de pions :

- Morpion : chaque joueur peut placer un pion sur une case qui n'est pas encore occupée.
- Puissance4 : chaque joueur peut placer un pion dans une colonne qui n'est as encore pleine. Le pion descend alors verticalement jusqu'à rencontrer un pion ou le bas de la colonne.

Voici deux configurations gagnantes, avec quatre pions bleus alignés :





Question 1. (2 points)
Définir le constructeur et la méthode *getCouleur()* de la classe *Pion*.

```
public class Pion {
  boolean couleur;
  public Pion(boolean c);
  public boolean getCouleur();
}
```

La classe *JeudePion* permet de mettre en facteur les fonctionnalités communes aux jeux de pions. La grille est un tableau à deux dimensions d'instances de la classe *Pion*, une absence de pion étant représentée par *null*. L'attribut *nbAlignes* a pour valeur le nombre de pions qu'il faut aligner pour obtenir une configuration gagnante.

```
abstract public class JeuDePion {
  private int nbAlignes;
  protected Pion[][] grille;
  public JeuDePion(int l, int c, int na)
  public boolean estLibre( int i, int j)
  abstract public void poserPion( int i, int j, boolean c)
              throws Exception
  private boolean estAligneH( int i, int j)
  private boolean estAligneV( int i, int j)
  private boolean estAligneD( int i, int j)
  private boolean estAligneAD( int i, int j)
  public boolean estAligne()
}
```

Question 2. (3 points)

Définir le constructeur de la classe *JeuDePion*, ainsi que la méthode *estLibre(int i, int j)* qui retourne *true* si la case de coordonnées (i, j) n'est pas occupée par un pion et *false* sinon.

Question 3. (4 points)

Définir la méthode *estAligneH(int i, int j)* qui retourne *true* s'il y a *nbAlignes* pions de la même couleur alignés horizontalement de façon contiguë, à partir de la case de coordonnées (i, j), et vers la droite.
En supposant connues les 3 autres méthodes *estAligneV*, *estAligneD*, et *estAligneAD*, définir la méthode *estAligne()* qui retourne *true* s'il y a un alignement de *nbAlignes* pions de la même couleur sur la grille.

Question 4. (2 points)

Définir les constructeurs des classes *Morpion* et *Puissance4*.

```
public class Morpion extends JeuDePion{
  public Morpion(int l, int c, int na)
  public void poserPion( int i, int j, boolean c)
    throws Exception
}
```

Question 5. (4 points)

Définir la méthode *poserPion* de la classe *Morpion*. La méthode modifie la grille en créant une nouvelle instance de *Pion* si la case n'est pas occupée, ou lève une exception si les indices ne sont pas corrects ou si la case est déjà occupée par un pion.

Question 6. (5 points)

Définir la méthode *poserPion* de la classe *Puissance4*. La méthode modifie la grille en créant une nouvelle instance de *Pion* dans la colonne de rang *j* si la colonne n'est pas pleine. Le pion est placé en haut de la colonne s'il reste une place libre, sinon une exception est levée.

```
public class Puissance4 extends JeuDePion {
  public Puissance4 (int l, int c, int na)
  public void poserPion(int i, int j, boolean c)
  throws Exception
}
```