Université Paris XII
IUT de Fontainebleau
Département informatique
Licence SIL
Structures de données abstraites
2006–2007

PARTIEL 1
Durée : 1h30

Seuls sont autorisés, à titre de documents, les listings comprenant explicitement le nom de l'étudiant imprimé (et celui-ci seulement) et les notes manuscrites (pas de photocopie) comprenant le nom de l'étudiant sur chaque page.

Il est conseillé d'écrire les exercices en langage C++. L'écriture en Java complique les entrées-sorties et l'écriture en langage C allonge le programme (ce qui ne rapportera pas de point supplémentaire dans un cas comme dans l'autre).

Exercice unique.- **Remplissage d'une région**

Un problème courant en graphisme est de **remplir une région** *: on a une image, on désigne un pixel de celle-ci et on veut que la région à laquelle appartient ce point change soit marquée. Dans le cas d'image monochrome, on changeait la couleur de celle-ci ; dans le cas d'image couleur, on indique une nouvelle couleur et toute la région doit prendre cette nouvelle couleur.*

À un moment ou un autre une **image** *sera un tableau bidimensionnel d'entiers : au pixel de coordonnées x et y de couleur c, on fera correspondre :*

$$tab[x, y] = c.$$

Qu'est-ce qu'une région ? Les spécialistes en sont venus à distinguer les 4-régions, les 8-régions ou d'autres notions. Dans le cas le plus simple des 4-régions, chaque pixel, de coordonnées (x, y), possède quatre voisins immédiats, ceux de coordonnées $(x, y-1)$ (au-dessus), $(x-1, y)$ (à gauche), $(x+1, y)$ (à droite) et $(x, y+1)$ (en-dessous), bien entendu que ces pixels appartiennent à l'image.

Considérons l'image suivante :

```
2 0 4 6 0
0 0 1 4 0
5 0 0 8 0
1 0 0 8 0
0 1 0 0 0
```

Remplir la région à laquelle appartient le pixel de coordonnées $(1, 1)$, d'ancienne couleur 0, avec la couleur 7 nous conduit à l'image suivante :

```
2 7 4 6 7
7 7 1 4 7
5 7 7 8 7
1 7 7 8 7
0 1 7 7 7
```

Remarquons que le pixel en bas à gauche n'a pas changé de couleur (il aurait changé de couleur avec les 8-régions).

L'algorithme pour résoudre ce problème utilise une pile pour garder trace des pixels qu'il faut exaaminer :

```
Empiler les coordonnées du point de départ.
Tant que la pile est non vide :
    dépiler un point P
    si la couleur de P est l'ancienne couleur
       empiler les (au plus) quatre voisins de P
       faire prendre à P la nouvelle couleur
```

- 1^o) Implémenter une classe pile de coordonnées (deux entiers).

- 2^o) Implémenter une classe `Image` dont les attributs seront un tableau bidimensionnel d'entiers (d'au plus 100×100 entiers), un entier `xmax` (le nombre maximun effectif de pixels horizontaux) et un entier `xmax` (le nombre maximum effectif de pixels verticaux) et dont les méthodes sont le constructeur par défaut, une méthode `lireImage()`, une méthode `afficheImage()` et une méthode `fill()` à quatre arguments entiers : les coordonnées du point déterminant la région, l'ancienne couleur et la nouvelle couleur.

La méthode `lireImage()` sera très rudimentaire : on demande à l'utilisateur les valeurs de `xmax`, `ymax` et des couleurs de chaque pixel un par un.

L'affichage ressemblera aux exemples ci-dessus : on affiche les couleurs de chacun des pixels de la première ligne (séparées de deux espaces), puis de la deuxième ligne et ainsi de suite..

- 4^o) Écrire un programme de test.