info301 : Algoritmique
TP 1, 2 et 3 : le jeu de UNO

Gwenael Gaborit et Pierre Hyvernat
Gwenael.Gaborit@univ-savoie.fr
Pierre.Hyvernat@univ-savoie.fr

Préliminaires

- Nous ne demandons pas de rapport : votre code doit contenir tout ce qui est nécessaire à la compréhension. (Il faut mettre des commentaires !)
- La répartition des taches sur les TP est purement informative : vous pouvez gérer votre temps comme vous l'entendez.
- Faites attention à la lisibilité de votre code : allez à la ligne, choisissez des noms de variables informatifs, mettez des commentaires...
- Votre programme final devra être envoyé par email à votre encadrant. Vous enverrez tous vos fichiers sources (et aucun fichier binaire). Deux détails :
  - votre programme devra compiler. Sans ça, votre note sera divisée par deux ! (Il peut compiler et ne pas fonctionner correctement, c'est un autre problème.)
  - Tous vos fichiers sources doivent comporter une entête où vous indiquerez vos noms et votre filière. Sans ça, votre note sera divisée par trois !
- Utilisez les commentaires pour préciser la structure de votre programme quand cela n'est pas clair ; mais ne mettez pas de commentaire du style "`je fais une boucle "for"`" : ça ne sert à rien.
- Vous pouvez contacter votre encadrant de TP par email, a condition de précisez dans le sujet qu'il s'agit du cours d'info301. (Sinon, le message risque d'être oublié dans un coin...)
- Au fait, si on ne vous l'a pas encore dit, commentez vos programmes... `:)`

Partie 0 : les règles du UNO

Le but de ce jeu est de se débarrasser de ses cartes (initialement sept par joueur) en jouant l'un après l'autre. Il y a deux types de cartes : les cartes normales caractérisées par un chiffre (0–9) et une couleur (rouge, vert, bleu ou jaune) ; et les cartes spéciales. À chaque tour, le joueur peut se séparer d'une carte normale uniquement si elle possède la même couleur ou le même chiffre que la précédente. Un joueur ne pouvant se débarrasser d'une carte doit en piocher une. Si elle est "jouable", celle-ci peut alors être jouée immédiatement.

Les cartes spéciales ont les effets suivants :
- "sauter le tour" : le joueur suivant passe son tour,
- "+2" et "+4" : le joueur suivant pioche deux ou quatre cartes et passe son tour,
- "changer de sens" : change le sens du jeu.

Ces cartes ont également une couleur et ne peuvent être jouées que sur la bonne couleur, ou sur une carte spéciale de même type.

Il y a en plus des "jokers" : ils peuvent être joués après n'importe quelle carte, et la couleur suivante est désignée par le joueur qui le pose.

Un joueur qui ne possède plus qu'une carte doit l'annoncer en disant "UNO". S'il l'oublie et qu'un autre joueur le remarque, il doit piocher deux cartes.

Les règles plus détaillées et les variantes du jeu sont consultables sur wikipedia :

`http://fr.wikipedia.org/wiki/Uno`

Le but de ce TP est de proposer une version informatique de ce jeu.

Partie 1 : les cartes (TP 1)

Question 1. En vous concertant avec votre binôme, définissez et choisissez un type de données "`carte`" pour représenter les carte d'un jeu de UNO.

Question 2. Vous définirez ensuite le jeu de cartes complet au sein d'un tableau, puis écrivez
- une procedure (ou une fonction) qui mélange le jeu,
- une procédure qui enregistre le jeu (mélangé ou non) dans un fichier,
- une procédure qui récupère le jeu dans un fichier.

Remarques :

À ce stade du TP, n'oubliez pas de tester vos procédures. Pour cela, écrivez une procédure d'affichage des cartes que vous pourrez utiliser pour afficher l'état du jeu (mélangé ou non). Lors de cette phase, vous pouvez vous contenter d'utiliser un jeu qui ne comporte qu'une dizaine de cartes.

La meilleure méthode pour mélanger un tableau de manière uniforme est d'utiliser l'algorithme suivant :

```
{***** procédure pour mélanger le tableau tab "en place" *****}
Randomize ;              {pour initialiser la "graine"}
for i:=N downto 2 do     {N est la taille du tableau}
begin
  u := Random(i)+1 ;     {u prend une valeur aléatoire entre 1 et i}
  tmp := tab[u] ;
  tab[u] := tab[i] ;
  tab[i] := tmp ;
end
```

Question 3. Chaque joueur sera représenté par la liste de ses cartes. On utilisera pour cela des listes simplement chaînées (cf TD).

Définissez le type de données "`joueur`" et écrivez les procédures permettant
- d'afficher les cartes,
- d'insérer une nouvelle carte,
- d'obtenir la valeur d'une carte dans la liste, (attention, on ne peut pas renvoyer une valeur de type enregistrement : il faut donc soit faire une fonction par champs, soit faire une procédure...)
- de jouer une carte (et donc de la supprimer).

Bien entendu, n'oubliez pas de tester vos fonctions et procédures !

Partie 2 : les joueurs (TP 2)

Question 1. Créez une unité intégrant les fonctions et procédures du TP 1 et utilisez une procédure principale pour tester cette unité.

Question 2. La liste des joueurs sera gérée par une liste circulaire* doublement chaînée pour pouvoir facilement changer le sens de jeu (carte "changer de sens").

Vous devez avoir vu cette structure de données en TD...

Dans votre unité, rajoutez le type de données approprié et écrivez les procédures suivantes :
- insérer un joueur,
- supprimer un joueur,
- afficher la liste des joueurs,
- passer au joueur suivant.

Testez toutes vos procédures sur des exemples.

---

* après un tour de jeu, c'est le premier joueur qui recommence

Partie 3 : le jeu et l'interface (TP 3)

Vous avez maintenant tous les ingrédients pour pouvoir jouer : réalisez un interface simple de jeu en mode texte, suffisante pour tester votre UNO.

Réfléchissez en particulier aux points suivants (mais n'essayez pas de tout implanter) :

- quand doit-on vérifier la validité des coups ?
- Est-il possible de tricher ?
- Comment gérer les règles officielles pour la carte "+4" (*cf.* l'article sur wikipedia),
- Que faire lorsque le talon est épuisé ?
- Comment gère-t'on la fin du jeu ?
- Comment compte-t'on les points ?
- Veut-on gérer les variantes des règles ?
- ...