Université de Nice–Sophia Antipolis
E.S.I.N.S.A. 2ème Année

février 2004

Projet d'informatique de 2ème année

Les sujets sont à réaliser en binôme. Chaque sujet est attribué à exactement 2 binômes. Vous êtes responsables de la constitution des binômes et de la répartition des sujets entre eux. Vous transmettrez cette répartition, fruit de vos négociations (toute manœuvre d'intimidation prohibée), à M. Granet dans **la semaine du 1er mars**.
Certains sujets font référence à des sites Internet, ou à des méthodes et algorithmes qui n'ont pas été présentés en cours. Si pour une raison quelconque vous aviez des difficultés à trouver l'information nécessaire pour réaliser votre projet, contactez vos enseignants au plus tôt :

`vincent.granet@unice.fr`
`ponsini@i3s.unice.fr`
`anne.vigouroux@unice.fr` (pour les sujets 2 et 7)

1 Sujets

Sujet 1 Visuel Karnaugh

Ce sujet aborde la simplification des fonctions booléennes à travers deux méthodes :
– les *tableaux de Karnaugh* sont une approche visuelle et intuitive de la simplification ;
– l'*algorithme de Quine – McCluskey*, quant à lui, apporte une solution plus systématique.
Votre travail consistera à mettre en œuvre ces deux méthodes en Java. Vous pourrez vous limiter à des fonctions de, au plus, 4 variables booléennes combinées avec les opérateurs logiques *et*, *ou* et *non*. Vous proposerez une interface graphique qui permet à l'utilisateur de remplir un tableau de Karnaugh (à la main dans un premier temps, puis automatiquement à partir de l'équation non simplifiée éventuellement en forme canonique) et de construire la fonction simplifiée. L'utilisateur pourra alors comparer sa solution avec celle obtenue par l'algorithme de Quine – McCluskey.

Note : Une description de cet algorithme peut être trouvée sur le site Internet (en anglais) :
`http://gene.bsee.swin.edu.au/dbraendler/QuineMcCluskey.htm` .

Sujet 2 Atelier d'optique géométrique

Il s'agit de réaliser un logiciel en Java qui permette de faire des exercices d'optique géométrique dont la résolution repose essentiellement sur des *tracés de rayon*. L'atelier disposera d'une bibliothèque de *systèmes optiques* de base (prismes, miroirs, lentilles) que l'utilisateur pourra sélectionner. Ensuite, le logiciel permettra de placer, aléatoirement ou non, un objet dont l'utilisateur doit déterminer l'emplacement de l'image en traçant des rayons lumineux. Le logiciel vérifiera la correction du tracé de l'utilisateur.

Sujet 3 Blokus

Blokus est un jeu de stratégie pour 2 ou 4 joueurs qui se joue sur une grille avec des pièces aux formes rectangulaires semblables à celles utilisées dans *Tetris*, un illustre prédécesseur. Le but de chaque joueur est de placer le plus possible de ses 21 pièces tout en empêchant les adversaires d'en faire autant.

Vous réaliserez un programme Java permettant à des joueurs humains de jouer ensemble à ce jeu. Dans un deuxième temps, vous offrirez aussi la possibilité de jouer contre l'ordinateur en mettant en œuvre un *algorithme de rétro-parcours*.

Note : Les règles complètes du jeu, ainsi qu'une *applet* de démonstration sont disponibles sur le site Internet :
`http://www.blokus.com` .

Sujet 4 Gomoku Ninuki

Ce jeu à deux, proche du *Morpion*, est compliqué par la possibilité de prendre des pions à son adversaire. Pour gagner, un joueur doit aligner sur une grille 5 de ses pions ou bien prendre 5 paires de pions à son adversaire.

Vous réaliserez un programme Java permettant à des joueurs humains de jouer ensemble à ce jeu. Dans un deuxième temps, vous offrirez aussi la possibilité de jouer contre l'ordinateur en mettant en œuvre un *algorithme de rétro-parcours*.

Note : Les règles complètes du jeu sont disponibles sur le site Internet :
`http://jeuxdesociete.free.fr/jeux/jeu-gomoku.html` .

Sujet 5 Robot programmable

Pour la première partie de ce projet, vous devrez simuler en Java le fonctionnement d'un robot dont le comportement pourra se limiter, mais vous êtes libres de l'enrichir, à quelques actions de déplacement (avancer, reculer, tourner, *etc.*), de détection (obstacle frontal, *etc.*) et de modification de son environnement (laisser une trace au sol, effacer une trace, *etc.*).

Vous devrez ensuite rendre le robot programmable : c'est-à-dire que l'utilisateur aura la possibilité d'écrire un programme (dans un langage de commande que vous définirez) qui sera chargé dans le robot pour être exécuté par ce dernier. Le programme sera une succession de commandes interprétables par le robot. Le langage de commande comprendra :
– toutes les actions définies dans la première partie, avec la possibilité de les faire exécuter séquentiellement ;
– au moins, une des extensions suivantes :
  – structure de contrôle conditionnelle (*ex. :* `si ... alors`, *etc.*),
  – structure de contrôle itérative (*ex. :* `répéter n fois`, `tantque ... faire`, *etc.*).

Note : Ce robot n'est pas sans rappeler la célèbre et ancestrale *tortue Logo* dont vous avez déjà peut-être entendu parler.

Sujet 6 Jeu de Go pédagogique

Dans le jeu de *Go*, deux joueurs s'opposent sur une grille avec des pions de couleur différente et tentent de s'approprier le plus grand territoire possible. Pour cela ils doivent constituer des *chaînes* avec leurs pions, tout en évitant de se les faire capturer. Les règles assez simples de ce jeu donnent naissance à une grande variété de situations complexes.

Dans le cadre de ce projet, vous réaliserez un programme Java d'initiation au jeu de *Go*. Vous proposerez une introduction progressive aux règles et situations typiques de ce jeu. Vous offrirez ensuite un mode de jeu pour que deux personnes puissent jouer l'une contre l'autre, avec éventuellement des fonctions d'assistance pour les novices.

Note : Les règles complètes du jeu sont disponibles, par exemple, sur le site Internet :
`http://jeudego.org` .

Sujet 7 Mesure de puissance en monophasé

L'objectif de ce projet est la réalisation en Java d'un TP virtuel dans lequel l'utilisateur doit établir le *bilan de puissance* d'un circuit électrique en effectuant des mesures de courants, tensions

et puissances en des points précis du circuit. Ensuite, l'utilisateur tracera le *diagramme de Fresnel* (tracé de vecteurs) sur l'écran par l'intermédiaire de la souris.

Sujet 8 Tan Gram

Le *Tan Gram* s'apparente à un puzzle dans lequel il faut agencer 7 pièces aux formes géométriques simples, toujours les mêmes, pour reconstituer une forme donnée (souvent un animal ou un personnage). Vous développerez en Java un programme permettant à l'utilisateur de manipuler les pièces du *Tan Gram* et de composer un puzzle. Vous fournirez une petite bibliothèque de puzzles à laquelle l'utilisateur pourra ajouter ses propres formes, ou une partie en cours, afin de les conserver entre deux exécutions du programme.

Par la suite, vous offrirez un mode de jeu dans lequel l'utilisateur tentera de résoudre un puzzle choisi parmi ceux de la bibliothèque. Le programme sera capable de détecter si la solution proposée par l'utilisateur résout effectivement le puzzle sélectionné.

Note : De nombreux sites Internet sont consacrés à ce jeu. En voici deux exemples :
- `http://tangrams.ca` (en anglais) et
- `http://echecsetmaths.ifrance.com/echecsetmaths/` (en français mais moins fourni).

Sujet 9 Spinout puzzle

Solitaire inspiré du *baguenaudier*, ce jeu se compose d'une règle où coulissent plusieurs pièces identiques attachées les unes aux autres. Une extrémité de la règle est fermée, l'autre est ouverte mais un butoir empêche les pièces de sortir lorsqu'elles sont en position verticale. Un espace aménagé dans la règle, une position avant le butoir, permet de faire passer une pièce de la position verticale à la position horizontale (et vice-versa). De plus, la forme des pièces empêche la pièce qui suit une pièce en position horizontale de pivoter. Le but du jeu est de placer toutes les pièces en position horizontale pour leur permettre de sortir de la règle (puis de les remettre toutes en position verticale pour retrouver la configuration d'origine).

Vous réaliserez un programme Java permettant à un joueur humain de jouer à ce jeu. Ensuite, vous présenterez graphiquement la solution de ce casse-tête qui repose sur le *code binaire de Grey*.

Note : Vous pourrez trouver des représentations de ce jeu sur Internet (par exemple deux sites en anglais) :
- `http://www.public.coe.edu/~jchoi//library/java/spin/` ou
- `http://www.geocities.com/jaapsch/puzzles/spinout.htm` .

2 Règles de contrôle du Projet

La note finale de projet d'informatique de 2ème année d'Esinsa est composée de deux notes attribuées :

1. **À la présentation orale**. Elle comprendra un exposé de 10 minutes au maximum, expliquant, à l'aide de transparents, les tenants et aboutissants du projet. Vu le peu de temps accordé, les orateurs devront faire preuve de synthèse. Cet exposé sera suivi d'une petite démonstration du logiciel. Exposer est un exercice difficile qui ne s'improvise pas. Répétez votre soutenance et préparez la démonstration de votre logiciel.
2. **Au rapport écrit**. Il comportera :
   - Une page de titre (titre du projet, noms des élèves, date...).
   - Une table de matières, comprenant les titres des chapitres, sections, sous-sections, avec les numéros des pages où ils apparaissent.
   - Une introduction qui explique clairement le sujet et où il vous conduit.
   - Un chapitre qui décrit le fonctionnement du logiciel, ce qu'il fait, comment on s'en sert.

   - Un chapitre qui décrit les principaux algorithmes et structures de données mis en jeu. C'est là que vous présenterez vos choix d'implémentation.
   - Une conclusion.
   - Une bibliographie. Vous donnerez ici les références des ouvrages qui vous ont aidés dans votre projet. Si dans le texte du rapport, vous citez une phrase d'un ouvrage (livre ou autre), vous devez lui associer la référence correspondante.
   - Le listing du programme. Celui-ci doit être correctement présenté, indenté, et clairement commenté.

Pour la frappe de votre rapport, vous utiliserez LaTeX.