Travaux Dirigés d'Algorithmique n°1

Cours d'Informatique de Deuxième Année

—DEUG SCIENCES S4-MI option Informatique—

Files - réalisation d'un nibbler

Cette séance de travaux dirigés est consacrée à l'utilisation d'une nouvelle structure de donnée : la file. Nous appliquons cette structure pour écrire un petit jeu, le nibbler.

On cherche à réaliser un programme de poursuite d'un ver s'allongeant au fur et à mesure qu'il consomme des proies.

Au lancement du programme un quadrillage régulier est affiché et des proies sont disséminées aléatoirement dans le quadrillage. Les proies sont représentées par des carrés pleins dans le damier.

Le ver est représenté par une tête (disque noir) et une suite d'anneaux représentés par des cercles. Au lancement du jeu, le ver est sans anneau. Ce ver est « piloté » par l'utilisateur en pointant avec la souris dans une case voisine de la tête. Le déplacement est simulé en déplaçant la tête dans la case pointée et en effaçant le dernier anneau de la queue.

Lorsque la tête arrive sur une case contenant une proie, celle-ci est consommée et un anneau supplémentaire est ajouté à l'extrémité de la queue.

Réalisation du damier

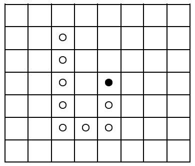
1. Écrire une fonction `Damier` qui affiche une fenêtre contenant un damier (on prendra par exemple un damier 20×20 dont le coté des case est de longueur 40) ;
2. écrire une fonction `Proies` qui dispose aléatoirement des proies sur le damier (par exemple 40 proies).

Gestion du ver

Nous proposons de gérer notre ver de la manière suivante :

- nous utilisons un tableau `Position` pour conserver les positions de notre ver. Quelle taille faut-il prévoir pour ce tableau ?

- les positions de notre ver occupent des cases contiguës de `Position`. Par exemple, pour la configuration suivante :



  on stocke les informations du ver dans :
  - un tableau de positions

4	5	6	7	8	9	10	11	12	13	14
...	$(2,3)$	$(3,3)$	$(4,3)$	$(5,3)$	$(6,3)$	$(6,4)$	$(6,5)$	$(5,5)$	$(4,5)$	...

    Quelle est la propriété des couples de ce tableau ?
  - deux variables `Tete` et `Queue` permettant de repérer les positions de la tête et de la queue dans le tableau (ici `Tete` $= 13$ et `Queue` $= 5$).
- La gestion d'un déplacement se fait simplement. Soit (i,j) la nouvelle position de la tête du ver (nous supposons cette position valide). Nous plaçons le couple (i,j) dans la case `Tete+1`. Comme nous venons de faire un déplacement de notre ver, la queue doit aussi bouger. Ainsi il faut incrémenter la valeur `Queue` de 1.

Implantation

Pour obtenir un programme modulaire avec un code clair et facilement modifiable, on suivra les consignes suivantes :

- on rassemblera les informations d'un objet dans une structure (coordonnées d'un point, description du ver ...) ;
- les différents états d'une case (vide, proie, ...) seront codés par un type `enum` et on évitera les constantes numériques dans le corps du programme (utiliser `#define`) ;
- les variables globales sont **INTERDITES** ;
- on veillera à la modularité du programme, aussi bien pour les fonctions liées à la gestion du ver, l'affichage du damier, la gestion de la souris ...

À vous de jouer : écrire un programme qui implante ce jeux (fin de la partie quand il n'y a plus de proie sur le damier, ou quand le ver se mord la queue).