Conventions de développement

Commentaires et longueurs maximums

Les commentaires sont à éviter sauf pour indiquer la/les précondition(s) d'une fonction. Dans ce cas, le commentaire est obligatoire. Il doit être inséré juste avant le nom de la fonction et commencer par PRECONDITION :

Une ligne de code ne doit pas dépasser 100 caractères.

Une fonction ne doit pas dépasser 150 lignes.

Indentation du code

L'accolade d'ouverture d'un bloc est placée sur une nouvelle ligne. L'accolade de fermeture de bloc est également placée sur une nouvelle ligne.

Dans le cas d'un appel d'une fonction avec de nombreux paramètres, s'il est nécessaire de placer les paramètres sur plusieurs lignes, ces paramètres sont indentés pour être positionnés au niveau du premier caractère situé après la parenthèse ouvrante de l'appel de la fonction.

Bon exemple :

```
/* PRECONDITION : 0<= origine.l <= 5 et 0<= origine.c <= 5 */

void calcule_cases_voisines (NUMBOX origine, NUMBOX** voisins,
                             int* taille)
{
	int i;

	*taille = 0;
	if (origine.l +1 <= 5) (*taille)++;
	if (origine.l -1 >=0) (*taille)++;
	if (origine.c +1 <= 5) (*taille)++;
	if (origine.c -1 >=0) (*taille)++;
	*voisins = malloc((*taille) * sizeof(NUMBOX));

	i=0;
	if (origine.l +1 <= 5)
	{
		(*voisins)[i].l = origine.l+1;
		(*voisins)[i].c = origine.c;
		i++;
	}
	if (origine.l -1 >= 0)
	{
		(*voisins)[i].l = origine.l-1;
		(*voisins)[i].c = origine.c;
		i++;
	}
	if (origine.c +1 <= 5)
	{
		(*voisins)[i].l = origine.l;
		(*voisins)[i].c = origine.c+1;
		i++;
	}
	if (origine.c -1 >= 0)
	{
		(*voisins)[i].l = origine.l;
		(*voisins)[i].c = origine.c-1;
		i++;
	}
}
```

Types

Toutes les variables qui ont comme valeurs possibles `vrai` ou `faux`, doivent être de type `BOOL` (ce type est définit dans la bibliothèque `graphics.h`) et non de type `int`.

Les noms des types que vous créez doivent être en majuscule. Le nom choisi doit caractériser la sémantique du type.

Exemple :

Un type structuré contenant deux entiers `x` et `y`, qui représente la position d'un point dans un repère orthonormé (`x` pour l'abscisse et `y` pour l'ordonnée) ne sera pas nommé `COUPLE`, mais `POINT`.

Nommage des fonctions

Le nom d'une fonction doit commencer par un verbe suivi du caractère '_' suivi d'au moins un mot précisant l'entité sur lequel le verbe s'applique.

Les mots qui forment le nom d'une fonction doivent tous être séparés par le caractère '_'

Exemple : affiche_carré, est_dans_carré, récupère_clic, …

Nommage des variables

Le nom d'une variable doit représenter la sémantique du contenu de celle-ci. Autrement dit, elle doit répondre à la question : quelle est l'information qui est contenue dans la variable ? On ne s'intéresse pas au type ici.

Les mots qui forment le nom d'une variable doivent tous être collés les uns aux autres. De plus, tous les mots à partir du second doivent commencer par une majuscule.

Les itérateurs des boucles for doivent être nommées `i`, puis `j`, puis `k` (boucle interne).

Exemple :

- Une variable qui contient le point en haut à gauche d'un carré ne s'appellera pas `P`, mais plutôt `hautGauche` ou plus simplement `hg`. En revanche, une variable que vous utilisez successivement pour contenir le point en haut à gauche d'un carrée, puis le centre d'un cercle pourra s'appeler `P`.
- Une variable qui contient la distance entre 2 points s'appellera `dist` et non `v`.
- Une variable qui contient la distance max entre 2 points parmi un ensemble de 10 points s'appellera `distMax`.