Examen

Exercice 1 (3 points) Donnez le code de la fonction `facto` transformant un nombre n en son factoriel $n! = n \times (n-1) \times \ldots \times 2$, par exemple on déclare `int x=4;` puis après l'appel on a $x = 24$, dans les cas suivants:
1) L'appel avait été effectué par l'instruction `facto(&x);`
2) L'appel avait été effectué par l'instruction `facto(x);`
3) L'appel avait été effectué par l'instruction `x=facto(x);`

Exercice 2 1) (4 points) Ecrire une classe cannonique `PointRn` simulant les points de $\mathbb{R}^n$ avec: deux attributs privés `int dim` et `double* adr`, un constructeur, un destructeur, un constructeur de recopie, et l'opérateur `=`.

2) (2 points) Munir la classe `PointRn` des opérateurs `==`, `!=`, `*`, `+` en tant que fonctions amies.

3) (2 points) Ecrire une classe abstraite `boule` avec: deux attributs privés `PointRn centre` et `double rayon`, et une fonction membre `bool contient(point p)`.

4) (3 points) Faire trois classes `boule1`, `boule2` et `bouleINF` dérivées de `boule` redéfinissant la méthode `contient` testant si le point en paramètre est ou non dans la boule, chacune des boules étant définie respectivement avec la norme L_1 ($\sum_i |x_i|$), la norme L_2 ($(\sum_i x_i^2)^{1/2}$) et la norme L_∞ ($\max_i |x_i|$). (On pourra supposer que `sqrt()` est disponible).

5) (3 points) Ecrire une fonction `bool intersecte(boule b1, boule b2)` renvoyant `false` si l'intersection des deux boules est vide.

Exercice 3 (3 points) Faire une fonction `int** arrangement(int m)` construisant la table des valeurs de

$$ar(n,p) := \binom{n}{p} \times p! = n \times \ldots \times (n-p+1)$$

pour $n = 0,1,\ldots,m$ et $0 \leq p \leq n$. Par exemple l'appel `arrangement(4)` construit un tableau :

1
1 1
1 2 2
1 3 6 6
1 4 12 24 24

On procédera **impérativement** de la façon décrite ci-dessous (enfin, si on veut avoir des points...):
1) On utilisera un tableau de tableaux via un pointeur de pointeurs `int** mat`. Le tableau de tableaux contient $m+1$ tableaux `mat[i]` de taille $i+1$ pour $i = 0,1,\ldots,m$. On fera l'allocation de mémoire avec $m+2$ appels à `new` et la désallocation avec $m+2$ appels à `delete[]`. Par exemple `int** mat = new int* [m+1];` pour le tableau de tableaux...

2) On initialisera `mat` de la façon suivante : `mat[i][0]=1` pour tout i. Puis on complètera `mat` avec la relation suivante $ar(n,p) = p \times ar(n-1,p-1) + ar(n-1,p)$ avec $ar(n-1,p) = 0$ pour $p = n$. Ainsi la valeur de `mat[i][j]` sera égale à $ar(i,j)$.

Problème 1 (2 points)

Un *amas* est une union de disques (boules du plan $\mathbb{R}^2$). Faire une classe canonique `amas` dérivée de `boule2` ayant en attribut privé `boule2 * am` pointant sur un tableau dynamique de boules. La munir d'une méthode `bool connexe(amas am)` testant si l'amas et connexe ou non.

Faire des classes équivalentes à `amas` pour les deux autres normes.