Exercice : Les sphères de collision

Dans la plupart des engins de physique utilisés en jeux vidéo, la gestion des collisions est approximée et non calculée de manière exacte. Il est en effet impossible de valider avec une précision parfaite qu’un projectile est entré en contact avec un personnage quelconque. **Il est nécessaire d’approximer cette détection**.

L’approximation peut être diverses. On utilise parfois un *volume englobant* pour approximer la forme d’un objet. Savez-vous pourquoi? Pouvez-vous nommer quelques-uns des volumes englobants possibles?

Dans le cadre de cet exercice, nous allons nous concentrer sur la codification de l’un de ces volumes englobants : les sphères.

1. Créez-vous un nouveau projet console et ajoutez-y un main vide.
2. Ajoutez au projet une classe (.h et .cpp) CollisionSphere et déterminez les propriétés, méthodes et constructeur(s) nécessaires.
3. Assurez-vous d’avoir une méthode vérifiant si la sphère est en collision avec une autre sphère reçue en paramètre. C’est à vous de déterminer l’algorithme nécessaire.
4. Ajoutez les instructions nécessaires dans le main pour :
   1. Saisir à la console les informations pertinentes pour créer les sphères de collision.
   2. Créer les sphères en question.
   3. Valider si elles entrent en contact et afficher le résultat à l’écran. N’oubliez pas de vous questionner sur ce qui arrive lorsque deux sphères se touchent sans se chevaucher. Répondez à cette question en commentaire dans votre code.
   4. Demander à l’utilisateur s’il souhaite recommencer ou non et terminer le programme le cas échéant.
5. Mettez le code du main actuel en commentaire et ajoutez-en un deuxième de manière à ce que 20 sphères soient créées au hasard dans une surface de 1360 x 720 unités. Assurez-vous de :
   1. Afficher les informations sur chaque sphère créée.
   2. Faire en sorte que les sphères ne débordent jamais au-delà de la surface.
   3. Le rayon de chaque sphère soit déterminé au hasard entre 20 et 200 unités.

**Note** : Le fait d’utiliser des concepts graphiques seulement en console n’est pas grave. Idéalement, les jeux bien construits au niveau de l’architecture ont une représentation abstraite complète qui peut fonctionner en dehors de l’affichage (disons que c’était vrai à une époque; avec les engins modernes de plus en plus complexes, tout est imbriqué)

Remettez votre projet sur LÉA avant dimanche soir (le 27 août à minuit). Assurez-vous d’avoir effacé tous les fichiers non nécessaires (voir la dernière diapositive du fichier Du\_C\_au\_C\_-\_partie\_2.pdf)