Exercices de C++ - jour 1

Rappels sur le C++ :

Exercice 1 :
Créez un namespace « Library » qui contiendra des classes permettant de gérer une bibliothèque. Écrivez une classe « Book » qui a un titre, un auteur, une année de publication et un statut de disponibilité (disponible ou indisponible).
Créez aussi une classe « Library » qui possède une liste de livres et un nom. Elle aura des méthodes permettant de lui ajouter un livre, d'afficher les livres disponibles, d'emprunter un livre en indiquant son nom (ce qui changera son statut), de rendre le livre en indiquant son nom, d'indiquer le nombre total de livre de la bibliothèque et son nombre total de livre disponible.
Pensez aussi à gérer les cas où un livre n'existe pas dans la bibliothèque et ne peut être rendu ou emprunté.

Utilisation avancée de l'héritage :

Exercice 2 :
Créez une classe *Shape* avec une méthode *calcArea()*. Puis créez une classe *Circle* dérivant de *Shape* avec sa propre méthode *calcArea()* qui calcule l'aire d'un cercle. Utilisez la clause « using » pour exporter la méthode *calcArea()* de la classe de base dans la classe dérivée et comparez les résultats.

Exercice 3 :
Reprenez l'exercice précédent puis développez une hiérarchie de classes représentant différentes formes géométriques : cercle, rectangle, triangle, carré et forme. Chaque forme doit avoir des propriétés spécifiques, une méthode pour calculer l'aire et une méthode pour calculer le périmètre.
Créez aussi un gestionnaire de formes qui peut stocker et gérer un ensemble de formes. Ce gestionnaire doit être capable de calculer l'aire totale de toutes les formes stockées.

Exercice 4 :
Vous développez un système de gestion des personnes pour une université. Créez les classes

suivantes :

- Une classe « personne » composée d'un nom, un prénom et une année de naissance.
- Une classe « étudiant » qui hérite de « personne » et qui dispose d'un numéro d'étudiant, d'une série de notes et d'une spécialisation. Une méthode permettra de calculer sa moyenne.
- Une classe « enseignant » héritant de « personne » avec une matière enseignée.
- Une classe « employé administratif » héritant aussi de « personne » et ayant un numéro d'employé ainsi qu'un poste.

Ajoutez ensuite une interface « travailleur » qui définit une méthode *travailler()* et qui sera implémentée par les étudiants, les enseignants et les employés administratifs.
Puis créez une classe « étudiant employé » qui hérite à la fois des classes « étudiant » et « employé administratif ». Cette classe doit aussi implémenter l'interface « travailleur ». Enfin testez votre code.

Les nouveautés de C++ 11 :

Exercice 5 :
Créez une fonction qui prend un pointeur en argument et affiche "Pointeur valide" s'il n'est pas nul et "Pointeur nul" s'il est nul. Utilisez *nullptr* pour vérifier si le pointeur est nul.

Exercice 6 :
Reprenez l'exercice 1 et modifiez la classe « Library » de façon à ce qu'elle ait un constructeur généré automatiquement, un constructeur qui prenne un nom en paramètre et initialise une liste de livres et un constructeur qui prenne un nom et une liste de livres. Mettez en place une délégation de constructeur. Affichez la liste des livres disponibles à travers une boucle sur un interval.

Exercice 7 :
En utilisant *std::transform*, créez une lambda qui calcule le carré des nombres d'un vecteur d'entiers.

Exercice 8 :
Modernisez le code C++ suivant sous forme d'objet de façon à respecter les standards actuels.

```cpp
#include <iostream>
#include <algorithm>

void printArray(int *arr, int size) {
    for (int i = 0; i < size; ++i) {
        std::cout << arr[i] << " ";
    }
    std::cout << std::endl;
}

void reverseArray(int *arr, int size) {
    for (int i = 0; i < size / 2; ++i) {
        int temp = arr[i];
        arr[i] = arr[size - i - 1];
        arr[size - i - 1] = temp;
    }
}

int main() {
    int size = 5;
    int *arr = new int[size];
    for (int i = 0; i < size; ++i) {
        arr[i] = i + 1;
    }

    std::cout << "Original array: ";
    printArray(arr, size);

    reverseArray(arr, size);

    std::cout << "Reversed array: ";
    printArray(arr, size);

    delete[] arr;
    return 0;
}
```