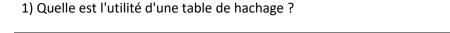


Merci de respecter les cadres pour les réponses. Tout débordement ne sera pas pris en compte et pourra être pénalisé de 1 point maximum par exercice.

<b>Question de cours</b>	(5 points)	/ 1pt par question
--------------------------	------------	--------------------



Stocker un grand nombre d'éléments et faire des recherches rapides (O(1))

2) Je souhaite associer des clés à des valeurs. Quel(s) conteneur(s) puis-je utiliser?

Une map ou unordered\_map (table de hachage). Accepter les deux ou alors un des deux NB : Mettre 0 à la question s'ils ont écrit autre chose (set ou vecteur ou list...)

3) A quoi sert la zone protégée dans une classe?

Donner un accès aux attributs aux classes filles

4) J'ajoute dans un tableau les entiers 5,2,3,3 et 5. Quel conteneur utiliser pour récupérer les nombres de cette manière : 2,3,5 ? Justifiez en une phrase.

Un set car tri automatique et suppression des doublons NB : Mettre 0 s'ils ont utilisé un autre conteneur...S

5) Donner 2 des 4 avantages d'une référence par rapport aux pointeurs.

Garantie que l'objet référencé est non nul / pas de fuite mémoire Plus facile à utiliser (pas de flèche, ni de déréférencement)



#### Exercice 1 : Diagramme de classe (4 points)

Considérons un jeu de dame simplifié représenté par une classe *Dame* contenant un ensemble de *pions* (20 pions noirs et 20 pions blancs), un tableau 2D comme plateau de jeu et un entier pour la taille du plateau de jeu. On souhaite gérer le déplacement et l'affichage des pions ainsi que la condition de victoire.

Faire le diagramme de classe en faisant apparaître pour chaque classe :

- 1. les attributs et leur visibilité
- 2. les méthodes et leur visibilité
- 3. les relations et les cardinalités entre les classes

Il n'est pas demandé de faire apparaître les constructeurs, destructeurs et getters/setters.

1) Diagramme de classe (3 points)

Noter au feeling : je demande que <u>deux</u> classes Dame et Pion, j'ai pénalisé s'il y a d'autres classes (exemple plateau / joueur)

2) Expliciter en français la relation liant les classes Dame et Pion (1 point)

Composition = relation forte si la classe Dame est détruite alors les pions sont détruits NB : Mettre 0 s'ils parlent d'héritage...



#### Exercice 2 : Compréhension de code de base (2 points)

```
#include <vector>
#include <iostream>
void swap(std::vector<char> tab, int a, int b) {
    char c = tab[a];
    tab[a] = tab[b];
    tab[b] = c;
}
void display(const std::vector<char>& tab) {
    for(unsigned int i = 0; i < tab.size(); i++) {</pre>
        std::cout << tab[i] << ",";
    std::cout << "END" << std::endl;</pre>
}
void mystery(const std::vector<char>& tab) {
    for (unsigned int i = 0; i < tab.size()-1; i++) {
        for(unsigned int j = i; j < tab.size(); j++) {</pre>
            if (j == i) min = j;
            if (tab[j] > tab[min]) min = j;
        swap(tab, i, min);
    }
}
int main() {
    std::vector<char> tab = {'a', 'b', 'd', 'c'};
    mystery(tab);
    display(tab);
    return 0;
```

Ecrire la sortie du programme (ce qui sera affiché à l'écran lors de son exécution). Justifiez!

```
Le fameux piège :
a b d c END car il manque une référence dans la fonction swap...
NB : 1 point pour la justification
```

#### Exercice 3 - Template et classe (4 points)

Ecrire une classe template nommée *Test* contenant une liste de type T *m\_array* et un entier *m\_size*. Vous <u>implémenterez</u> le constructeur surchargé, le destructeur ainsi qu'un accesseur en lecture permettant de récupérer la liste de type T. **(3 points)** 

};		

Donner un exemple d'instanciation de votre classe pour les types std::string et float. (1 point)

```
Test<std ::string> obj(...);
Test<float> obj2(...);

NB : Accepter si les paramètres ne sont pas donnés dans le constructeur
```

#### Exercice 4 - Héritage et polymorphisme (5 points)

Considérons une classe mère Vehicule ainsi que trois classes filles Moto, Camion et Voiture.

Toutes ces classes possèdent une méthode afficher().

On souhaite gérer un garage contenant un ensemble de véhicules.

1) Ecrire la classe Garage contenant un attribut m\_vehicules ainsi qu'une méthode afficher. (1 point)

```
Class Garage
{
    Private :
        Std ::vector<Vehicule*> m_vehicules ; // accepter std ::list mais pas d'accès avec [] !
    Public :
        Garage() ;
        "Garage() ;
        Void afficher() ; // pas de virtual !
} ;

NB : Un vecteur de référence n'existe pas ! Donc -0,5pt
Si pas de pointeurs, le polymorphisme ne fonctionne pas donc -0.5pt
```

2) Implémenter la méthode *afficher* de la classe *Garage*. Cette méthode affichera les informations des véhicules (en utilisant la méthode *afficher* des classes filles déjà fournie) **(2 points)** 

```
Void Garage ::afficher()
{
    For (const auto& voiture : m_vehicules)
    {
        Voiture->afficher() ; // attention c'est une flèche car pointeur !
    }
}
```



3) Que peut-on dire de la méthode afficher de la classe mère ? (1 point)

Elle est virtuelle <u>pure</u> car inutile dans la classe mère Vehicule.
NB : Puisque ce n'était pas très clair dans l'énoncé, j'ai retiré 0.5pt si on me dit que c'est virtuel.

4) Que peut-on dire de la classe Vehicule ? (1 point)

D'après la question 3, c'est donc une classe abstraite
D apres la question 3, c est donc dife classe abstraite

Bon courage!