**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.**

**Question de cours (5 points)**

1) Quelle est la différence entre une classe et un objet ?

|  |
| --- |
| La classe est un nouveau type de variable  L’objet est une instance d’une classe |

2) Qu'est-ce que le principe d'encapsulation ?

|  |
| --- |
| Les attributs sont en private. L’utilisateur doit donc passer par des getters et setters pour les modifier ou y accéder. |

3) Qu'est-ce qu'une table de hachage ?

|  |
| --- |
| Une table de hachage est un conteneurs qui permet de stocker un grand nombre d’entrée. Elle prend une entrée et la transforme à l’aide d’une fonction de hash ce qui permet de récupérer l’identifiant de la valeur est de le stocker dans le conteneur. |

4) Quelle est la différence entre une pile et une file ?

|  |
| --- |
| Une pile est selon le principe LIFO = Last In First Out.  Une file est selon le principe FILO = First In Last Out. |

5) Donner 2 relations possibles entre des classes. Expliquer en une phrase ces relations.

|  |
| --- |
| Composition : Relation forte entre deux objets, si l’un est détruit l’autre l’est également.  Agrégation : relation faible entre deux objets, si l’un est détruit l’autre ne l’est pas. |

**Exercice 1 : Etude d'un texte (3 points)**

*Dans un parking se trouve de nombreux véhicules de différentes couleurs et tailles. On peut dénombrer 10 voitures, 3 motos et 2 camions de 10 tonnes. Contrairement aux véhicules légers, les camions peuvent transporter jusqu'à 100m3 de marchandises. Titounette est le nom de ma voiture, Titine est celui de la moto de mon amie...*

Identifier sous forme de tableau : les classes, objets et attributs

|  |  |  |
| --- | --- | --- |
| *Classes*  Véhicules  Voitures  Moto  Camions | *Objets*  Titounette  Titine | *Attributs*  Couleurs  Tailles  Place de stockage |

**Exercice 2 (5 points)**

Soit la classe suivante dont le rôle et de remplir aléatoirement N entiers dans un vecteur, d’afficher ses valeurs et de calculer la moyenne. N sera fourni par l'utilisateur.

class Dummy  
{  
 private:  
 std::vector<int> m\_tab;  
 int m\_N; // nombre d’entiers du vecteur  
 public:  
 Dummy(int N);   
 ~Dummy();  
 void remplirAlea(); // remplir aléatoirement le vecteur  
 void afficher(); // afficher les valeurs du vecteur  
 float moyenne(); // retourner la moyenne du vecteur  
};

1) Implémenter en C++ le constructeur

|  |
| --- |
| Dummy::Dummy(int N) : m\_N(N)  {} |

2) Implémenter en C++ la méthode remplirAlea()

|  |
| --- |
| Void Dummy :: remplirAlea(){  For(int i=0; i<m\_N; i++){  M\_tab.push\_back(rand%500);  }  } |

3) Implémenter en C++ la méthode moyenne()

|  |
| --- |
| Float dummy :: moyenne(){  Float value(0.0) ;  For(int i=0 ; i<m\_tab.size(); i++){  Value+=tab[i];  }  Return Value/m\_tab.size();  } |

4) Implémenter en C++ la méthode afficher()

|  |
| --- |
| Void dummy ::afficher(){  For(int i=0 ; i<m\_tab.size(); i++){  Std::cout << “value at “ << I << “ is “ << tab[i] << std::endl;  }  } |

5) Ecrire les lignes du main qui appellent les 3 méthodes ci-dessus. L’affichage de la moyenne se fera dans le main

|  |
| --- |
| Int main(){  Dummy Obj ;  Obj.remplirAlea() ;  Obj.afficher() ;  Std ::cout << « moyenne = «  << Obj.moyenne() << std ::endl ;  Return 0 ;  } |

**Exercice 3 : (2 points)**

class A  
{  
 public:  
 A() { std::cout << "Contructeur A" << std::endl; }  
 ~A() { std::cout << "Destructeur A" << std::endl; }  
};  
  
class B  
{  
 public:  
 B() { std::cout << "Contructeur B" << std::endl; }  
 ~B() { std::cout << "Destructeur B" << std::endl; }  
 void test() { std::cout << "TEST" << std::endl; }  
};  
  
int main()  
{  
 A\* a = new A();  
 B b;  
 b.test();  
 return 0;  
}

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

|  |
| --- |
| Constructeur A  Constructeur B  TEST  Destructeur B |

**Exercice 4 (5 points)**

1) Ecrire une classe mère *Personnage* comportant deux attributs : *m\_point\_vie* et *m\_point\_defense*

|  |
| --- |
|  |

2) Ecrire une classe *Magicien* héritant de *Personnage* ayant en plus comme attribut : *m\_point\_magie*

|  |
| --- |
|  |

3) Ecrire deux classes *MagicienBlanc* et *MagicienNoir* héritant de *Magicien*

|  |
| --- |
|  |

4) Donner l'ordre d'appel des constructeurs quand je créé un objet de type *MagicienBlanc*.

|  |
| --- |
|  |

5) Soit p1 un objet de type *Personnage* et mg1 un objet de type *MagicienNoir*. Peut-on faire l'instruction suivante : "mg1 = p1;" ? Expliquer brièvement pourquoi.

|  |
| --- |
|  |

Bon courage !