Examen

Durée 2 heures - Sujet recto - Aucun document ni ordinateur n'est autorisé.

Exercice 1 (14 points)

On considère une classe « Entier » représentant un nombre entier et dont l'attribut permettant de stocker la valeur entière **devra être alloué dynamiquement**. La déclaration de la classe est la suivante.

```
class Entier
{
public:
    Entier();         // Constructeur par défaut (init. à 0)
    Entier(int v);
    ~Entier();
    Entier(const Entier&);

    Entier operator=(Entier& e);
    Entier operator=(int e);
    Entier& operator+=(Entier& e);

    friend ostream& operator<<(ostream&, Entier&);

private:
    int* valeur;
};
```

1. Proposez un programme principal utilisant toutes les capacités de cette classe. Créez 2 objets dont l'un est alloué statiquement et l'autre dynamiquement.
2. Ecrivez le code de chacune des méthodes figurant dans la déclaration.

Exercice 2 (6 points)

On dispose d'une classe « Compteur » dont la déclaration de la classe et la définition des méthodes sont données ci-dessous.

```
class Compteur
{
public:
    Compteur();
    void inc();
    void dec();
    int get_valeur();

private:
    int valeur;
};
```

```
Compteur::Compteur(){
    valeur = 0;
}

void Compteur::inc(){
    valeur++;
}

void Compteur::dec(){
    valeur--;
    if (valeur<0)
        valeur = 0;
}

int Compteur::get_valeur(){
    return valeur;
}
```

On souhaite maintenant définir une classe « Compteur_borne » héritant de la classe « Compteur », et dont l'incrémentation de la valeur ne pourra excéder une valeur limite, qui devra être un attribut de la classe et initialisé par le constructeur.

1. Écrire la classe « Compteur_borne » (déclaration de classe et code des méthodes) permettant d'obtenir ce comportement.