Class Personne

```
class Personne
{
    // Attributs
    private string prenom;
    private string nom;
    private int age;
    static long nbPersonne;

    //Constructeurs
    public Personne(string P, string N, int A)
    {
        nbPersonne++;
        Prenom = P;
        Nom = N;
        Age = A;
        Console.WriteLine("Constructeur personne (string,string,int)");
    }

    public Personne(Personne p)
    {
        nbPersonne++;
        Prenom = p.prenom;
        Nom = p.nom;
        Age = p.age;
        Console.WriteLine("Constructeur enseignant (Personne)");
    }

    //Propriétés
    public string Prenom
    {
        get { return prenom; }
        set
        {
            if (value == null || value.Trim().Length == 0)
            {
                throw new Exception("prénom (" + value + ") invalide!");
            }
            else
            {
                prenom = value;
            }
        }
    }
    //Propriétés
    public string Nom
    {
        get { return nom; }
        set
        {
            if (value == null || value.Trim().Length == 0)
            {
                throw new Exception("nom (" + value + ") invalide!");
            }
            else
            {
```

```
                    nom = value;
                }
            }
        }
        //Propriétés
        public int  Age
        {
            get { return age; }
            set
            {
                if (value < 0)
                {
                    throw new Exception("age (" + value + ") invalide!");
                }
                else
                {
                    age = value;
                }
            }
        }
        public static long NbPersonne
        {
            get { return nbPersonne; }
        }
        //Ou public string Identite
        public virtual string Identite
        {
            get { return String.Format("[{0},{1},{2}]", prenom, nom, age); }
        }

        public override string ToString()
        {
            return Identite;
        }
    }
```

Class Enseignant

```
    class Enseignant: Personne
    {
        private int section;

        public Enseignant(string P, string N, int A, int S)
            : base(P, N, A)
        {
            section = S;
            Console.WriteLine("Constructeur enseignant (string,string,int,int)");
        }

        //Si Identite dans Personne est non virtual:
        //public new string Identite
        public override string Identite
        {
            get { return String.Format("Enseignant[{0},{1}]", base.Identite, section); }
        }
        public override string ToString()
        {
            return Identite;
```

```
        }

    }
```

Class ListeDePersonne

```
    class ListeDePersonne : ArrayList
    {
        public static ListeDePersonne operator +(ListeDePersonne l, Personne p)
        {
            l.Add(p);
            return l;
        }
        public override string ToString()
        {
            string listeToString = "(";
            for (int i = 0; i < Count - 1; i++)
            {
                listeToString += this[i].Identite;
                listeToString += ",";
            }
            if (Count != 0)
                listeToString += this[Count - 1].Identite;
            listeToString += ")";
            return listeToString;
        }
        public new Personne this[int i]
        {
            get { return (Personne)base[i]; }
            set { base[i] = value; }
        }
        public int this[string nom]
        {
            get
            {
                for (int i = 0; i < Count; i++)
                {
                    if (((Personne)base[i]).Nom == nom)
                        return i;
                }
                return -1;
            }
        }
    }
```