

1 Class Personne

```
1 class Personne
2 {
3     // Attributs
4     private string prenom;
5     private string nom;
6     private int age;
7     static long nbPersonne;
8
9     //Constructeurs
10    public Personne(string P, string N, int A)
11    {
12        nbPersonne++;
13        Prenom = P;
14        Nom = N;
15        Age = A;
16        Console.WriteLine("Constructeur personne (string,string,int)");
17    }
18
19    public Personne(Personne p)
20    {
21        nbPersonne++;
22        Prenom = p.prenom;
23        Nom = p.nom;
24        Age = p.age;
25        Console.WriteLine("Constructeur enseignant (Personne)");
26    }
27
28    //Propriétés
29    public string Prenom
30    {
31        get { return prenom; }
32        set
33        {
34            if (value == null || value.Trim().Length == 0)
35            {
36                throw new Exception("prénom (" + value + ") invalide!");
37            }
38            else
39            {
40                prenom = value;
41            }
42        }
43    }
44    //Propriétés
45    public string Nom
46    {
47        get { return nom; }
48        set
49        {
50            if (value == null || value.Trim().Length == 0)
51            {
52                throw new Exception("nom (" + value + ") invalide!");
53            }
54            else
55            {
```

```

56         nom = value;
57     }
58 }
59 }
60 //Propriétés
61 public int Age
62 {
63     get { return age; }
64     set
65     {
66         if (value < 0)
67         {
68             throw new Exception("age (" + value + ") invalide!");
69         }
70         else
71         {
72             age = value;
73         }
74     }
75 }
76 public static long NbPersonne
77 {
78     get { return nbPersonne; }
79 }
80 //Ou public string Identite
81 public virtual string Identite
82 {
83     get { return String.Format("[{0},{1},{2}]", prenom, nom, age); }
84 }
85
86 public override string ToString()
87 {
88     return Identite;
89 }
90 }

```

91 Class Enseignant

```

1     class Enseignant: Personne
2     {
3         private int section;
4
5         public Enseignant(string P, string N, int A, int S)
6             : base(P, N, A)
7         {
8             section = S;
9             Console.WriteLine("Constructeur enseignant (string,string,int,int)");
10        }
11
12        //Si Identite dans Personne est non virtual:
13        //public new string Identite
14        public override string Identite
15        {
16            get { return String.Format("Enseignant[{0},{1}]", base.Identite, section); }
17        }
18        public override string ToString()
19        {
20            return Identite;

```

```
21     }
22
23 }
```

24 Class ListeDePersonne

```
1     class ListeDePersonne : ArrayList
2     {
3         public static ListeDePersonne operator +(ListeDePersonne l, Personne p)
4         {
5             l.Add(p);
6             return l;
7         }
8         public override string ToString()
9         {
10            string listeToString = "(";
11            for (int i = 0; i < Count - 1; i++)
12            {
13                listeToString += this[i].Identite;
14                listeToString += ",";
15            }
16            if (Count != 0)
17                listeToString += this[Count - 1].Identite;
18            listeToString += ")";
19            return listeToString;
20        }
21        public new Personne this[int i]
22        {
23            get { return (Personne)base[i]; }
24            set { base[i] = value; }
25        }
26        public int this[string nom]
27        {
28            get
29            {
30                for (int i = 0; i < Count; i++)
31                {
32                    if (((Personne)base[i]).Nom == nom)
33                        return i;
34                }
35                return -1;
36            }
37        }
38    }
```