Solution Contrôle N° 1 (POO - Java)

1. **(0,75 point)**

```
Appart appart = new Appart();
```

2. **(1,5 point)**

```
Lampe li , lt, ll;
li = new Lampe (1, 75);
lt = new Lampe (2, 45);
ll = new Lampe (3, 15);
```

3. **(3 points)**

```
Piece piece1 = new Piece ();
piece1.setNature(1);
piece1.addLampe(li);
appart.addPiece(piece1);

Piece piece2 = new Piece ();
piece2.setNature(2);
piece2.addLampe(lt);
appart.addPiece(piece2);

Piece piece3 = new Piece ();
piece3.setNature(3);
piece3.addLampe(ll);
appart.addPiece(piece3);
```

4. **(0,75 point)**

```
appart.affListePieces();
```

B. **(1,5 points)**

```
Pièce utilisée en tant que Chambre
Pièce utilisée en tant que Cuisine
Pièce utilisée en tant que Salle de bain
```

C.

1. **(1 point)**

```
public int getConsom () {
   return consommation;
```

```
}
```

2. **(1 point)**

```
public int nbreLampes () {
   return lampes.size();
}
```

3. **(1 point)**

```
public void affListeLampes () {
   for (int i=0; i<lampes.size(); i++) {
      System.out.println(lampes.elementAt(i));
   }
}
```

D.

1. **(0.5 point)**

Elle calcule la consommation totale des lampes d'une pièce

2. **(1 point)**

Les éléments de la classe Vector sont des références de type Object. Donc, la méthode elementAt() retourne une référence sur des instances de type Object. Dans ce cas, la référence l est du type Lampe qui est une sous classe de Object. Le transtypage est nécessaire car une référence d'une sous classe ne peut pas référencer une instance d'une classe mère. Or, comme on sait que les instances référencées dans le vecteur lampes sont effectivement de type Lampe, on force ce transtypage tout en étant sûr qu'il est normal.

E. **(1,5 points)**

```
public void affDetail () {
   System.out.println("Nombre de lampes: " + nbreLampes());
   affListeLampes();
   System.out.println("Consommation totale: " + consomTotale() + " Watts");
}
```

F. **(1,5 point)**

```
public void affListePieces () {
   for (int i=0; i<pieces.size(); i++) {
      System.out.println(pieces.elementAt(i));
      ((Piece)pieces.elementAt(i)).affDetail();        }
```

```
}
```

G **(2 points)**

Pièce utilisée en tant que Chambre
Nombre de lampes: 1
Lampe: Incandescence, consommation: 75 Watts
Consommation totale: 75 Watts
Pièce utilisée en tant que Cuisine
Nombre de lampes: 1
Lampe: Tube, consommation: 45 Watts
Consommation totale: 45 Watts
Pièce utilisée en tant que Salle de bain
Nombre de lampes: 1
Lampe: Led, consommation: 15 Watts
Consommation totale: 15 Watts

Exercice (3 points)

```
class Compte {
   private int numero; private float solde;
   private static int comptes=0;
   public void Compte(){ comptes++; }
   public void Compte(int numero, float solde){
         this.numero = numero;
         this.solde = solde;
         this.comptes++;
   }
   public void getSolde() { return solde; }
   public String getNumero () { return numero; }
   public void debiter(float M){
         if (M <= solde) solde -= M;
   }
   public String toString() { return numero + " " + solde; }
   public static void main (String[] args) {
          Compte c1 = new Compte(14, 7000), c2 = new Compte();
          System.out.println ("Compte 1 & 2: " + c1.numero + "/" + c2.numero);
          Compte c3, c4 = new Compte(12);
          if (c3.getSolde() == 0) System.out.println ("Le compte 3 est vide !");
      System.out.println ("Compte 4 créé sans solde !");
   }
}
```