

## Exercices dictionnaire commande

### Exercice 2 :

panier.Add("orange", 10); donne

« orange »
10

### Exercice 3 :

- « Le dictionnaire a 4 éléments »
- « orangecitronpommepoire »
- 5163
- Console.WriteLine(panier[quantite.GetEnumerator()] " : "+quantite+"kg\n");

### Exercice 5 :

Aperçu du **Commande.cs** :

```
using System;
using System.Collections.Generic;
using System.Text;

namespace dictionnaire
{
    class Commande
    {
        int id;
        DateTime dateCreation;
        Dictionary<Produit, Int32> lignes;
        public Commande(int i, DateTime d)
        {
            id = i;
            dateCreation = d;
            lignes = new Dictionary<Produit, Int32>();
        }
        public void ajouterLigne(Produit unProd, int uneQte)
        {
            if (lignes.ContainsKey(unProd))
            {
                lignes[unProd] = uneQte;
            } else
            {
                lignes.Add(unProd, uneQte);
            }
        }
        public void supprimerLigne(Produit unProd)
        {
            if (lignes.ContainsKey(unProd))
            {
                lignes.Remove(unProd);
            }
        }
        public Dictionary<Produit, Int32> GetLignes()
        {
            return lignes;
        }
        public void afficherCommande()
        {
            string msg = "";
            msg += "Num commande ";
            msg += this.id.ToString();
        }
    }
}
```

```

        msg += "\n";
        msg += "du : ";
        msg += this.dateCreation.ToString();
        msg += "\n";
        foreach (KeyValuePair<Produit, Int32> kvp in lignes)
        {
            msg += kvp.Key.Ref.ToString() + " ";
            msg += kvp.Key.Designation + " ";
            msg += kvp.Value;
            msg += "\n";
        }
        Console.WriteLine(msg);
    }
}

```

Aperçu de **Produit.cs** :

```

using System;
using System.Collections.Generic;
using System.Text;

namespace dictionnaire
{
    class Produit
    {
        private string _ref;
        private string _designation;
        private int _qteRestante;
        private double _prix;
        public Produit()
        {
            this._ref = "";
            this._designation = "";
            this._qteRestante = 0;
            this._prix = 0;
        }
        public Produit(string id, string libelle, int qte, double prix)
        {
            this._ref = id;
            this._designation = libelle;
            this._qteRestante = qte;
            this._prix = prix;
        }

        public string Ref { get => _ref; set => _ref = value; }
        public string Designation { get => _designation; set => _designation = value; }

        public int QteRestante { get => _qteRestante; set => _qteRestante = value; }
        public double Prix { get => _prix; set => _prix = value; }
    }
}

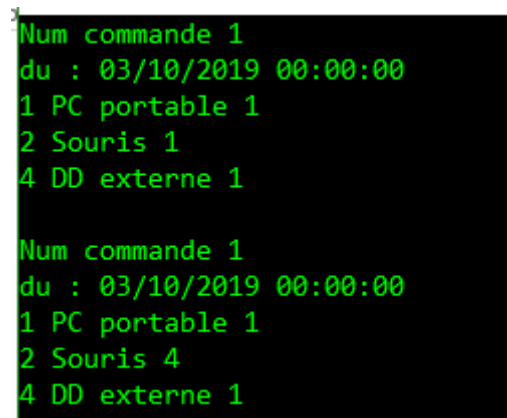
```

Aperçu de Program.cs :

```
using System;
using System.Collections.Concurrent;
using System.Collections.Generic;

namespace dictionnaire
{
    class Program
    {
        static void Main(string[] args)
        {
            Commande C1 = new Commande(1, DateTime.Today);
            Produit p1 = new Produit("1", "PC portable", 150, 299);
            // créer la classe produit
            Produit p2 = new Produit("2", "Souris", 200, 15);
            Produit p3 = new Produit("3", "Cle USB", 180, 14);
            Produit p4 = new Produit("4", "DD externe", 100, 75);
            C1.ajouterLigne(p1, 1);
            C1.ajouterLigne(p2, 1);
            C1.ajouterLigne(p4, 1);
            C1.afficherCommande(); // a ecrire
            C1.ajouterLigne(p2, 4); // finalement on veut commander 4 souris au lieu
de 1 souris initialement
            C1.afficherCommande();
            Console.ReadLine();
        }
    }
}
```

Aperçu du jeu d'essai :



```
1
Num commande 1
du : 03/10/2019 00:00:00
1 PC portable 1
2 Souris 1
4 DD externe 1

Num commande 1
du : 03/10/2019 00:00:00
1 PC portable 1
2 Souris 4
4 DD externe 1
```

### Exercice 6 :

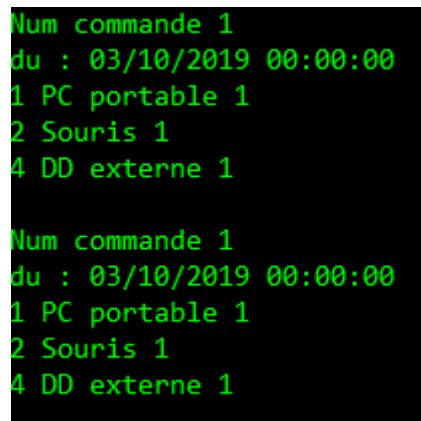
On veut maintenant que si le produit figure déjà sur la commande, que l'on garde la quantité d'origine entrée précédemment.

```
public void ajouterLigne(Produit unProd, int uneQte)
{
    if (lignes.ContainsKey(unProd))
    {
        // rien ne se passe, on conserve la quantité d'origine
    } else
    {
        lignes.Add(unProd, uneQte);
    }
}
```

Aperçu du `Program.cs` :

```
static void Main(string[] args)
{
    Commande C1 = new Commande(1, DateTime.Today);
    Produit p1 = new Produit("1", "PC portable", 150, 299);
    // créer la classe produit
    Produit p2 = new Produit("2", "Souris", 200, 15);
    Produit p3 = new Produit("3", "Cle USB", 180, 14);
    Produit p4 = new Produit("4", "DD externe", 100, 75);
    C1.ajouterLigne(p1, 1);
    C1.ajouterLigne(p2, 1);
    C1.ajouterLigne(p4, 1);
    C1.ajouterLigne(p2, 4);
    C1.afficherCommande();
    C1.ajouterLigne(p2, 10);
    C1.afficherCommande();
    Console.ReadLine();
}
```

Jeu d'essai :



```
Num commande 1
du : 03/10/2019 00:00:00
1 PC portable 1
2 Souris 1
4 DD externe 1

Num commande 1
du : 03/10/2019 00:00:00
1 PC portable 1
2 Souris 1
4 DD externe 1
```

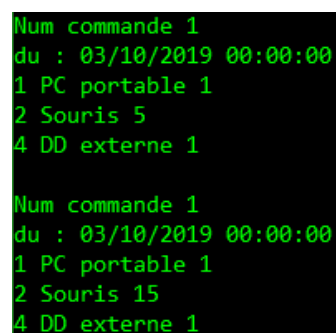
La quantité initiale est conservée

### Exercice 7 :

On veut maintenant que les quantités de produits commandés s'additionnent.

```
public void ajouterLigne(Produit unProd, int uneQte)
{
    if (lignes.ContainsKey(unProd))
    {
        lignes[unProd] += uneQte;
    } else
    {
        lignes.Add(unProd, uneQte);
    }
}
```

Le `Program.cs` n'a pas été modifié, je lance donc le programme et j'obtiens :



```
Num commande 1
du : 03/10/2019 00:00:00
1 PC portable 1
2 Souris 5
4 DD externe 1

Num commande 1
du : 03/10/2019 00:00:00
1 PC portable 1
2 Souris 15
4 DD externe 1
```