uses a compound orderby to sort a list of products, first by category, and then by unit price, from highest to lowest.

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

public class MainClass {
public static void Main() {

List products = GetProductList();
var sortedProducts =
from p in products
orderby p.Category, p.UnitPrice descending select p;

foreach (var s in sortedProducts) {
Console.Write(s);
}
}
static List GetProductList() {
List empTree = new List();
empTree.Add(new Product { ProductName = “A”, Category = “O”,
UnitPrice = 12, UnitsInStock = 5, Total = 36, OrderDate = new
DateTime(2005, 1, 1), Id = 1 });
empTree.Add(new Product { ProductName = “B”, Category = “O”,
UnitPrice = 2, UnitsInStock = 4, Total = 35, OrderDate = new
DateTime(2005, 1, 1), Id = 1 });
empTree.Add(new Product { ProductName = “C”, Category = “O”,
UnitPrice = 112, UnitsInStock = 3, Total = 34, OrderDate = new
```

```
DateTime(2005, 1, 1), Id = 1 });
empTree.Add(new Product { ProductName = “D”, Category = “O”,
UnitPrice = 112, UnitsInStock = 0, Total = 33, OrderDate = new
DateTime(2005, 1, 1), Id = 1 });
empTree.Add(new Product { ProductName = “E”, Category = “O”,
UnitPrice = 1112, UnitsInStock = 2, Total = 32, OrderDate =
new DateTime(2005, 1, 1), Id = 1 });
empTree.Add(new Product { ProductName = “F”, Category = “O”,
UnitPrice = 11112, UnitsInStock = 0, Total = 31, OrderDate =
new DateTime(2005, 1, 1), Id = 1 });
return empTree;
}
}
class Product : IComparable {
public string ProductName { get; set; }
public string Category { get; set; }
public int UnitPrice { get; set; }
public int UnitsInStock { get; set; }
public int Total { get; set; }
public DateTime OrderDate { get; set; }
public int Id { get; set; }

public override string ToString() {
return String.Format(“Id: {0}, Name: {1} , Category: {3}”,
this.Id, this.ProductName, this.Category);
}
int IComparable.CompareTo(Product other) {
if (other == null)
return 1;
if (this.Id > other.Id)
return 1;

if (this.Id < other.Id) return -1; return 0; } } [/csharp]
```