Universidad de la Habana, Facultad de Matemática y Computación

Carrera Ciencia de la Computación, Asignatura Programación

3er Problema de Programación, Curso 2007 – 2008

Implemente una DLL `Agrupador.DLL` que incluya en el namespace `Problema3` una clase `Agrupador` (sin constructor o con un constructor sin parámetros) que implemente la interface `IAgrupador`. Todas las interfaces que usted necesitará estarán en el ensamblado `Interfaces.DLL` en el namespace `Problema3`

```
interface IAgrupador{
  public IEnumerable Agrupa(IEnumerable coleccion, ISelector selector);
}
interface ISelector{
   object Selecciona(object x);
}
interface IGrupo: IEnumerable{
   object Llave{ get; }
}
```

La interfaz `ISelector` a través del método `Selecciona` permite obtener una llave a partir de un objeto cualquiera. Por ejemplo, a partir de una cadena de caracteres, la llave u objeto que devuelve el método `Selecciona` puede ser un objeto de tipo `char` (el primer carácter de la cadena), o de tipo `string` (la cadena pero toda en mayúsculas), o de tipo `int` (la longitud de la cadena), o incluso puede devolver el propio objeto string. La idea es que un conjunto de objetos que compartan un mismo valor de llave (según el método `Selecciona`) se puedan "agrupar" por este valor.

La interfaz `IGrupo` permite representar "grupos" de objetos donde cada grupo se caracteriza porque todos los objetos del grupo tienen una misma llave obtenida por el método `Selecciona`. La propiedad `Llave` nos da la llave que es común a todos los objetos que pertenecen al grupo. La colección de los objetos de cada grupo se puede obtener iterando al grupo (porque fíjese que un `IGrupo` tiene que implementar la interfaz `IEnumerable`).

- Al aplicar el método `Agrupa` (de su implementación de la interface `IAgrupador`) este debe devolver un `IEnumerable` formado a partir del parámetro `coleccion`. El `IEnumerable` resultante es un `IEnumerable` de objetos de tipo `IGrupo`.
- En el resultado no puede haber dos `IGrupos` con un mismo valor para la propiedad `Llave`.
- En cada objeto `IGrupo` deben estar todos los objetos de la colección original que ofrecen el mismo valor de llave al utilizar el método `Selecciona` de la instancia de `selector` pasada como segundo parámetro del método `Agrupa`.

- Cada elemento de la `coleccion` original debe aparecer en un único grupo (y si está repetido en la `coleccion` original debe aparecer repetido en el grupo en la misma cantidad en que aparece en la `coleccion` original).
- No puede haber un elemento en un `IGrupo` que no sea de la `coleccion` original

NOTA:

Fíjese que su DLL `Agrupador.DLL` respuesta no tiene por qué incluir una implementación de la interfaz `ISelector` pero sí tiene que incluir una implementación de la interface `IGrupo` ya que esta la necesitará para poder implementar su interface `IAgrupador`. (Aunque para comprobar su solución seguramente tendrá que crear alguna clase que implemente `ISelector`).

Considere que el método `Selecciona` del objeto `selector` que se le pasará como parámetro en las pruebas es aplicable a los objetos del `IEnumerable coleccion`.

Tenga en cuenta que el tipo del objeto que el método `Selecciona` devuelve como resultado puede ser de un tipo que implemente su propia definición de igualdad (redefine el `Equals`)

Ejemplo 1

Si se tiene una clase

```
class Persona{
   public string nombre;
   public Fecha fechaNacimiento;
   …
}
class Fecha{
  public int dia;
  public int mes;
  public int año;
  public override string ToString(){ ... }
  ...
}
```

donde `estudiantes` es una variable de tipo `IEnumerable` de personas que debe devolver la secuencia

```
{“juan”, {12, 2,1990}},     {“luis”, {20, 3,1993}},    {“ana”, {2,2,1990}},
{“marina”, {10,5,1992}},    {“jorge”,{1, 9, 1991}},    {“carmen”, {20,8,1991}},
{“pablo”, {30, 12, 1991}},  {“raisa”, {15, 11, 1992}}
```

y `SeleccionadordeAño` está implementado de la siguiente forma (note que se asume en este caso que el método `Selecciona` se le aplica a un parámetro `object` que será de tipo `Persona`)

```
class SeleccionadordeAño: ISelector{
  public object Selecciona(object x){
      Persona p = (Persona)x;
      return p.fechaNacimiento.año;
  }
}
```

Entonces, si su implementación de `IAgrupador` es la clase `MiAgrupador`, si se hace

```
IAgrupador a = new Problema3.MiAgrupador();
IEnumerable gruposEstudiantes = a.Agrupa(estudiantes, new SeleccionadordeAño());
foreach (IGrupo g in gruposEstudiantes) {
//cada g tiene una propiedad Llave y es a su vez IEnumerable de personas
    Console.WriteLine("\n{0}",g.Llave);
    foreach (Persona p in g)
        Console.WriteLine("  {0} {1}", p.nombre, p.fechaNacimiento);
}
```

se listaría como resultado

```
1990
  juan 12/2/1990
  ana 2/2/1990

1993
  luis 20/3/1993

1991
  jorge 1/9/1991
  carmen 20/8/1991
  pablo 30/12/1991

1992
  marina 10/5/1992
  raisa 15/11/1992
```

Nota:
El orden en que aparecen los grupos no es determinante.
Ni los grupos ni sus elementos tienen por qué quedar ordenados de ninguna forma en particular, ya que los tipos de los elementos y de las llaves ni siquiera están obligados a definir un criterio de orden (con

independencia de que en el ejemplo se han empleado llaves de tipo `int` que casualmente son ordenables pero note que los grupos de años no han tenido por qué quedar en orden según el año).

Ejemplo 2

Si para el mismo ejemplo anterior se le pasa como seleccionador un objeto del tipo

```
class SeleccionadorLongitud: ISelector{
  public object Selecciona(object x){
      Persona p = (Persona)x;
      return p.nombre.Length;
  }
}
```

Si se llama

```
IEnumerable gruposEstudiantes = a.Agrupa(estudiantes, new SeleccionadorLongitud());
```

en este caso los grupos se formarían según la longitud del nombre, quedando en cada grupo todas las personas que tienen nombres con la misma longitud. El resultado de ejecutar

```
foreach (IGrupo g in gruposEstudiantes){
    Console.WriteLine("\n{0}",g.Llave);
    foreach (Persona p in g)
        Console.WriteLine("  {0} {1}", p.nombre, p.fechaNacimiento);
}
```

sería entonces (note que en cada grupo todos son nombres que tienen la misma longitud)

```
4
  juan 12/2/1990
  luis 20/3/1993

3
  ana 2/2/1990

6
  marina 10/5/1992
  carmen 20/8/1991

5
  jorge 1/9/1991
  pablo 30/12/1991
  raisa 15/11/1992
```