Appello TAP del 13/01/2020

Scrivere nome, cognome e matricola sul foglio protocollo, indicando anche se avete nel piano di studi TAP da 6 CFU (quello attuale) o da 8 CFU (quello "vecchio"). Avete a disposizione tre ore.

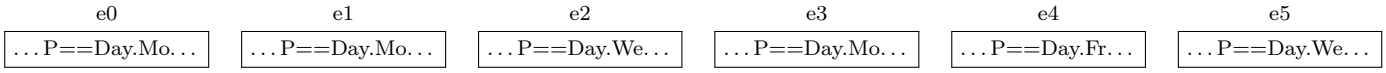
Esercizio 1 (6 punti)

Data l'interfaccia

```
public interface I<T> { T P { get; } }
```

Scrivere l'extension-method `Indexing` che, dato un array `s` di elementi di tipo `I<T>`, dove `T` è un parametro di tipo **istanziabile solo su un tipo elencazione**, produce il dizionario che associa ogni valore `v` di tipo `T` alla sottosequenza di `s` degli elementi di `s` **al momento della chiamata** che hanno `v` come valore della proprietà `P`.

Ad esempio, usando il tipo elencazione dei giorni lavorativi (`enum Day { Mo, Tu, We, Th, Fr}`), se `arr` è l'array `[e0, e1, e2, e3, e4, e5]`, dove



si avrà

```
var res = arr.Indexing();
for (int i = 1; i < arr.Length; ++i) arr[i] = arr[0];
// even changing arr is still true that
// res[Day.Mo] == [e0, e1, e3]
// res[Day.Tu] empty array
// res[Day.We] = [e2, e5]
// res[Day.Th] empty array
// res[Day.Fr] = [e4]
```

Il metodo dovrà sollevare `ArgumentNullException` se `s` o uno dei suoi elementi è `null`.

Hint: la classe `System.Enum`, da cui derivano i tipi elencazione, fornisce un metodo statico `GetValues` che dato un valore di tipo `Type` che rappresenta un tipo elencazione ne restituisce gli elementi. *Attenzione:* il tipo di ritorno di `GetValues` è `Array` (NON generico), le cui istanze sono collezioni di `object`.

Esercizio 2 ([1+4+4] = 9 punti)

Implementare, usando NUnit, i seguenti test relativi a `Indexing`, dell'esercizio 1.

1. Input della chiamata sotto test: `s` deve essere un array (non nullo) di 5 elementi di cui il terzo sia nullo

   Output atteso: eccezione di tipo `ArgumentNullException`

2. Esempio presentato nell'esercizio 1.

3. Test parametrico con parametro intero `howMany`, basato su un tipo elencazione di colori con 3 valori per bianco, grigio e nero.

   Input della chiamata sotto test: un array di dimensione 3*`howMany` in cui si ripetono ciclicamente elementi aventi la proprietà `P` dei 3 colori richiesti

   Output atteso: un dizionario che associa a ciascun colore un array di dimensione `howMany` contenente gli stessi oggetti che compaiono nell'input aventi la proprietà `P` di quel colore, nello stesso ordine in cui compaiono nell'input.

Esercizio 3 ([3+2] = 5 punti)

- Nella seguente classe `Node` (che realizza la classica lista "linkata" semplice), implementare il metodo `IsCircular` che verifica se la lista è una lista circolare.

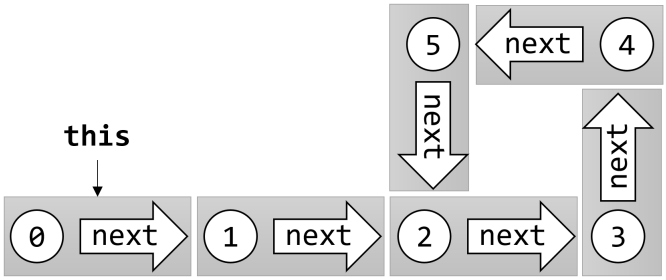
```
public class Node {
    public int I{ get; }
    public Node Next{ get; set; }
    public Node(int i, Node next){
        I = i;
        Next = next;
    }

    public bool IsCircular(){
        ...
    }
}
```

  Il metodo dovrà restituire `true` se e solo se navigando a partire da `this` lungo la catena dei `Next` si ritorna a `this`.

  Attenzione a eventuali "loop interni", in cui un `Next` riporta a un elemento intermedio della lista.

- Implementare, usando NUnit, il seguenti test relativi a `Next`.

  Input della chiamata sotto test:



  Output atteso: `false`.