```
public static class ZipClass
    {
        public static IEnumerable<T[]> Zip<T>(this IEnumerable<T>[]? s)
        {
            IEnumerable<T[]> Zip_Aux()
            {
                var size = s.Length;
                var sEnumerator = new IEnumerator<T>[size - 1];
                try
                {
                    for (int i = 0; i < size - 1; i++)
                    {
                        sEnumerator[i] = s[i + 1].GetEnumerator();
                    }

                    var a = AllMoveNext();
                    while (a > 0)
                    {
                        if (a != size) throw new ArgumentNullException("a");
                        var item = new T[size];
                        for (int i = 0; i < size; i++)
                          item[i] = sEnumerator[i].Current;
                        yield return item;
                        a = AllMoveNext();
                    }
                }
                finally
                {
                    for (int i = 0; i < size-1; i++)
                    {
                        sEnumerator[i].Dispose();
                    }
                }

                int AllMoveNext()
                {
                    var result = 0;
                    foreach (var enumerator in sEnumerator)
                    {
                        result += enumerator.MoveNext() ? 1 : 0;
                    }

                    return result;
                }
            }

            if (null == s) throw new ArgumentNullException(nameof(s));
            return Zip_Aux();
        }
    }
```

!!!!NO TEST!!!!

!!!!NO TEST!!!!

!!!!NO TEST!!!!