hashset.cm

```
/*
    ==============================================================================

    Copyright (c) 2012-2016 Seppo Laakko
    http://sourceforge.net/projects/cmajor/

    Distributed under the GNU General Public License, version 3 (GPLv3).
    (See accompanying LICENSE.txt or http://www.gnu.org/licenses/gpl.html
        )

 ==============================================================================
    */

// Copyright (c) 1994
// Hewlett-Packard Company
// Copyright (c) 1996
// Silicon Graphics Computer Systems, Inc.
// Copyright (c) 2009 Alexander Stepanov and Paul McJones

using System;
using System.Concepts;

namespace System.Collections
{
    public class HashSet<T, H = Hasher<T>, C = EqualTo<T>> where T is
        Semiregular and HashFunction<H, T> and C is Relation and C.Domain
        is T
    {
        public typedef T ValueType;
        public typedef T KeyType;
        public typedef H HashFun;
        public typedef C Compare;
        public typedef HashSet<ValueType, HashFun, Compare> Self;
        public typedef Hashtable<KeyType, ValueType, Identity<ValueType>,
             HashFun, Compare> TableType;
        public typedef TableType.ConstIterator ConstIterator;
        public typedef TableType.Iterator Iterator;

        public nothrow Iterator Begin()
        {
            return table.Begin();
        }
        public nothrow ConstIterator Begin() const
        {
            return table.CBegin();
        }
        public nothrow ConstIterator CBegin() const
        {
            return table.CBegin();
        }
        public nothrow Iterator End()
    {
        return table.End();
    }
    public nothrow ConstIterator End() const
    {
        return table.CEnd();
    }
    public nothrow ConstIterator CEnd() const
    {
        return table.CEnd();
    }
    public nothrow inline int Count() const
    {
        return table.Count();
    }
    public nothrow inline bool IsEmpty() const
    {
        return table.IsEmpty();
    }
    public nothrow void Clear()
    {
        table.Clear();
    }
    public nothrow Iterator Find(const KeyType& key)
    {
        return table.Find(key);
    }
    public nothrow ConstIterator Find(const KeyType& key) const
    {
        return table.CFind(key);
    }
    public nothrow ConstIterator CFind(const KeyType& key) const
    {
        return table.CFind(key);
    }
    public Pair<Iterator, bool> Insert(const ValueType& value)
    {
        return table.Insert(value);
    }
    public nothrow void Remove(const KeyType& key)
    {
        table.Remove(key);
    }
    public nothrow void Remove(Iterator pos)
    {
        table.Remove(pos);
    }
    private TableType table;
}

public nothrow bool operator==<T, H, C>(const HashSet<T, H, C>& left,
     const HashSet<T, H, C>& right) where T is Semiregular and
    HashFunction<H, T> and C is Relation and C.Domain is T
    {
        if (left.Count() != right.Count()) return false;
        for (const T& value : left)
        {
            if (right.Find(value) == right.End()) return false;
        }
        return true;
    }
}
```