Lesson 4:

Solution 1: FrogRiverOne

using System;

// you can also use other imports, for example:

// using System.Collections.Generic;

// you can write to stdout for debugging purposes, e.g.

// Console.WriteLine("this is a debug message");

class Solution {

public int solution(int X, int[] A) {

bool[] has = new bool[X];

int count = 0;

for (var i = -1; ++i < A.Length;) {

if (!has[A[i] - 1]) {

has[A[i] - 1] = true;

if (++count == X)

return i;

}

}

return -1;

// write your code in C# 6.0 with .NET 4.5 (Mono)

}

}

Exercise 2:

MaxCounters

|  |
| --- |
| var counters = new int[N]; |
|  | int max = 0, min = 0; |
|  | foreach (var n in A) |
|  | if (n == N + 1) |
|  | min = max; |
|  | else { |
|  | var v = counters[n - 1]; |
|  | if((counters[n - 1] = Math.Max(v, min) + 1) > max) |
|  | max = counters[n - 1]; |
|  | } |
|  | for (int i = counters.Length; --i >= 0; ) |
|  | if(counters[i] < min) |
|  | counters[i] = min; |
|  | return counters; |

Exercise 3:

MissingInteger

|  |
| --- |
| int min = 1; |
|  | HashSet<int> numbers = new HashSet<int>(); |
|  | foreach (int n in A) { |
|  | if (!numbers.Contains(n)) { |
|  | numbers.Add(n); |
|  | if (n == min) |
|  | while (numbers.Contains(++min)); |
|  | } |
|  | } |
|  | return min; |

Exercise 4:

PermCheck

using System;

using System.Collections.Generic;

// you can also use other imports, for example:

// using System.Collections.Generic;

// you can write to stdout for debugging purposes, e.g.

// Console.WriteLine("this is a debug message");

class Solution {

public int solution(int[] A) {

uint

expected = (uint)((A.Length + ((A.Length + 1) % 2)) \* Math.Ceiling(A.Length / 2.0)),

sum = 0;

HashSet<int> digits = new HashSet<int>(A);

if (digits.Count != A.Length)

return 0;

for (int i = -1; ++i < A.Length; sum += (uint)A[i])

;

return sum == expected ? 1 : 0;

// write your code in C# 6.0 with .NET 4.5 (Mono)

}

}