# 41

using System;

using System.Linq;

using System.Collections.Generic;

namespace plus

{

internal class Program

{

private static void Main(string[] args)

{

List<string> results = new List<string>();

int n = int.Parse(Console.ReadLine());

for (int i = 0; i < n; i++)

{

string text = Console.ReadLine().Trim(); // 移除字符串两端的空格

int index = 0;

string pattern = "";

string ans = "";

// 寻找第一个数字

while (index < text.Length && char.IsDigit(text[index]))

{

pattern += text[index];

index++;

}

ans += "LineNo=" + pattern + "\n";

pattern = string.Empty;

// 寻找冒号和空格之间的字符串（不含空格）

while (index < text.Length && !char.IsLetter(text[index]))

{

index++;

}

while (index < text.Length && char.IsLetter(text[index]))

{

pattern += text[index];

index++;

}

ans += "OP=" + pattern + "\n";

pattern = string.Empty;

// 寻找空格后面的字符串

while (index < text.Length && text[index] == ' ')

{

index++;

}

while (index < text.Length && (char.IsLetter(text[index]) || char.IsDigit(text[index])))

{

pattern += text[index];

index++;

}

ans += "N1=" + pattern;

pattern = string.Empty;

// 寻找后续字符串

while (index < text.Length && !char.IsLetter(text[index]) && !char.IsDigit(text[index]))

{

index++;

}

if (index != text.Length)

{

while (index < text.Length && (char.IsLetter(text[index]) || char.IsDigit(text[index])))

{

pattern += text[index];

index++;

}

ans += "\nN2=" + pattern;

}

results.Add(ans);

}

foreach (string result in results)

{

Console.WriteLine(result);

}

}

}

}

# 42五

# 43五

# 47 48 49

using System;

using System.Linq;

namespace A

{

class Program

{

static void Main(string[] args)

{

// 输入集合A的元素

int n = int.Parse(Console.ReadLine());

int[] setA = new int[n];

string[] setAElements = Console.ReadLine().Split(' ');

for (int i = 0; i < n; i++)

{

setA[i] = int.Parse(setAElements[i]);

}

// 输入集合B的元素数量

int m = int.Parse(Console.ReadLine());

int[] setB = new int[m];

string[] setBElements = Console.ReadLine().Split(' ');

for (int i = 0; i < m; i++)

{

setB[i] = int.Parse(setBElements[i]);

}

/\* // 计算并输出集合A与集合B的交集

int[] intersection = GetIntersection(setA, setB);

Console.WriteLine(string.Join(" ", intersection));\*/

/\*

int[] unionArr = GetUnion(setA, setB);

Console.WriteLine(string.Join(" ", unionArr));\*/

int[] exceptArr = GetExcept(setA, setB);

Console.WriteLine(string.Join(" ", exceptArr));

}

static int[] GetIntersection(int[] setA, int[] setB)

{

int[] intersection = new int[Math.Min(setA.Length, setB.Length)];

int intersectionSize = 0;

foreach (int elementA in setA)

{

foreach (int elementB in setB)

{

if (elementA == elementB)

{

intersection[intersectionSize] = elementA;

intersectionSize++;

break;

}

}

}

Array.Resize(ref intersection, intersectionSize);

return intersection;

}

static int[] GetUnion(int[] setA, int[] setB)

{

int[] unionArr = new int[setA.Length + setB.Length];

int unionSize = 0;

foreach (int elementA in setA)

{

unionArr[unionSize] = elementA;

unionSize++;

}

foreach (int elementB in setB)

{

if (!unionArr.Contains(elementB))

{

unionArr[unionSize] = elementB;

unionSize++;

}

}

Array.Resize(ref unionArr, unionSize);

return unionArr;

}

static int[] GetExcept(int[] setA, int[] setB)

{

int[] exceptArr = new int[setA.Length];

int Size = 0;

foreach (int elementA in setA)

{

if (!setB.Contains(elementA))

{

exceptArr[Size] = elementA;

Size++;

}

}

Array.Resize(ref exceptArr, Size);

return exceptArr;

}

}

}

# 53

public class Rect

{

private double x;

private double y;

private double l;

private double w;

public Rect(double x, double y, double l, double w)

{

this.x = x;

this.y = y;

this.l = l;

this.w = w;

}

public Rect(Rect r)

{

this.x = r.x;

this.y = r.y;

this.l = r.l;

this.w = r.w;

}

public double getX()

{

return x;

}

public void setX(double x)

{

this.x = x;

}

public double getY()

{

return y;

}

public void setY(double y)

{

this.y = y;

}

public double getL()

{

return l;

}

public void setL(double l)

{

this.l = l;

}

public double getW()

{

return w;

}

public void setW(double w)

{

this.w = w;

}

public double area()

{

return l \* w;

}

public int CompareArea(Rect r)

{

if (this.area() > r.area())

{

return 1;

}

else if (this.area() == r.area())

{

return 0;

}

else

{

return -1;

}

}

public string toString()

{

return "x:" + x.ToString("F5") + "\ny:" + y.ToString("F5") + "\nL:" + l.ToString("F5") + "\nW:" + w.ToString("F5") + "\n面积:" + area().ToString("F5");

}

}

# 56

interface IReadBook

{

void ReadBook();

}

class CBauStu : IReadBook

{

public void ReadBook()

{

Console.WriteLine("本科生读教材");

}

}

class CGduStu : IReadBook

{

public void ReadBook()

{

Console.WriteLine("硕士生读中文学术期刊");

}

}

class CDocStu : IReadBook

{

public void ReadBook()

{

Console.WriteLine("博士生读外文学术期刊");

}

}