实例7：输出狼人杀游戏主要角色

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("狼人杀游戏主要身份：");//提示信息

//定义数组，存储狼人杀游戏主要角色

string[] roles = { "狼人", "预言家", "村民", "女巫", "丘比特", "猎人", "守卫" };

foreach (string role in roles)

{

Console.Write(role+" ");

}

Console.ReadLine();

}

}

}



上机3：foreach语句应用

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

double[] points = { 80, 88, 86, 90, 75, 5 };

double sum = 0;

double avg = 0;

foreach (double point in points)

{

sum += point;

}

avg = sum / points.Length;

Console.WriteLine("总成绩为："+sum);

Console.WriteLine("平均成绩为："+avg);

Console.ReadLine();

}

}

}



5.4.1 数组的遍历

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

//存放字母个数

int Letters = 0;

//存放数字个数

int Digits = 0;

//存放标点符号个数

int Punctuations = 0;

//用户提供的输入

string instr;

Console.WriteLine("请输入一个字符串");

instr = Console.ReadLine();

//声明foreach循环以遍历输入的字符串中的每个字符

foreach (char ch in instr)

{

//检查字母

if (char.IsLetter(ch))

{

Letters++;

}

//检查数字

if (char.IsDigit(ch))

{

Digits++;

}

//检查标点符号

if (char.IsPunctuation(ch))

{

Punctuations++;

}

}

Console.WriteLine("字母个数为：{0}",Letters);

Console.WriteLine("数字个数为：{0}",Digits);

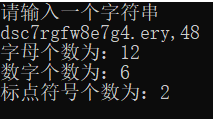
Console.WriteLine("标点符号个数为：{0}",Punctuations);

Console.ReadLine();

}

}

}



上机4：冒泡排序法应用

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

int[] a = { 5, 1, 7, 2, 3 };

for (int i = 0; i < a.Length; i++)

{

for (int j = 0; j < a.Length-i-1; j++)

{

if (a[j]>a[j+1])

{

int temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

Console.WriteLine("升序排序后的结果为：");

foreach (int b in a)

{

Console.Write(b+" ");

}

Console.ReadLine();

}

}

}



实例8：sort方法应用

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

int[] a = { 5, 3, 2, 4, 1 };

Array.Sort(a);

Console.WriteLine("排序后的结果为：");

foreach (int b in a)

{

Console.Write(b+" ");

}

Console.ReadLine();

}

}

}



实例1：从控制台输入一个字符串， 输出该字符串的长度，已经字符串中的第一个字符和最后一个字符

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

Console.WriteLine("字符串的长度为："+str.Length);

Console.WriteLine("字符串中第一个字符为："+str[0]);

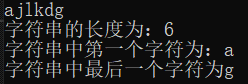
Console.WriteLine("字符串中最后一个字符为" + str[str.Length-1]);

Console.ReadLine();

}

}

}



实例2 ：逆序输出字符串中的字符

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

for (int i = str.Length-1; i >= 0; i--)

{

Console.WriteLine(str[i]);

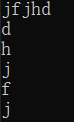
}

Console.ReadLine();

}

}

}



实例3：IndexOf用法

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

string str = "We are the world";//创建字符串

int firstIndex = str.IndexOf("r");//获取字符串中“r”第一次出现的索引位置

//获取字符串中“r”第二次出现的索引位置，从第一次出现的索引位置之后开始查找

int secondIndex = str.IndexOf("r", firstIndex + 1);

//获取字符串中“r”第三次出现的索引位置，从第二次出现的索引位置之后开始查找

int thirdIndex = str.IndexOf("r", secondIndex + 1);

//输出三次获取的索引位置

Console.WriteLine("r第一次出现的索引位置是："+firstIndex);

Console.WriteLine("r第二次出现的索引位置是：" + secondIndex);

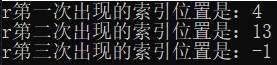
Console.WriteLine("r第三次出现的索引位置是：" + thirdIndex);

Console.ReadLine();

}

}

}



实例4：然后判断字符串中是否含有@，并输出@的位置

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

if (str.IndexOf("@") != -1)

{

Console.WriteLine("字符串中含有@，其出现的位置是{0}", str.IndexOf("@") + 1);

}

else

{

Console.WriteLine("字符串中不含有@");

}

Console.ReadLine();

}

}

}



实例5：判断该字符串中是否仅含有一个@

using System;

namespace PPT23test

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

int firstIndex = str.IndexOf("@");

int lastIndex = str.LastIndexOf("@");

if (firstIndex != -1)

{

if (firstIndex == lastIndex)

{

Console.WriteLine("在该字符串中仅含有一个@");

}

else

{

Console.WriteLine("在该字符串中含有多个@");

}

}

else

{

Console.WriteLine("字符串中不含有@");

}

Console.ReadLine();

}

}

}





