例:程序bit\_shift\_test1

// Copyright 2016.刘珅珅

// author：刘珅珅

// 位移测试

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace bit\_shift\_test1

{

class BitShitfTest

{

static void Main()

{

int val = 1;

for (int i = 0; i < 8; ++i)

{

for (int t = 128; t > 0; t = t / 2)

{

if ((val & t) != 0)

{

Console.Write("1 ");

}

if ((val & t) == 0)

{

Console.Write("0 ");

}

}

Console.WriteLine();

val = val << 1; // 左移

}

Console.WriteLine();

// 负数右移

// 负数右移时，在左边高位补1

// 1 1 1 1 1 1 1 1 0 1 1

// 1 1 1 1 1 1 1 1 1 0 1

// 1 1 1 1 1 1 1 1 1 1 0

// 1 1 1 1 1 1 1 1 1 1 1

// 1 1 1 1 1 1 1 1 1 1 1

// 1 1 1 1 1 1 1 1 1 1 1

// 1 1 1 1 1 1 1 1 1 1 1

// 1 1 1 1 1 1 1 1 1 1 1

val = -5;

for (int i = 0; i < 8; ++i)

{

for (int t = 1024; t > 0; t = t / 2)

{

if ((val & t) != 0)

{

Console.Write("1 ");

}

if ((val & t) == 0)

{

Console.Write("0 ");

}

}

Console.WriteLine();

val = val >> 1;

}

Console.WriteLine();

// 负数左移，在右边低位补0

// 在移动的过程中，有可能把负数移成正数

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1

// val is -5

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0

// val is -10

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0

// val is -20

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0

// val is -40

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0

// val is -80

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0

// val is -160

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0

// val is -320

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0

// val is -640

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0

// val is -1280

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0

// val is -2560

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0

// val is -5120

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0

// val is -10240

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0

// val is -20480

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -40960

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -81920

// 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -163840

// 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -327680

// 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -655360

// 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -1310720

// 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -2621440

// 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -5242880

// 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -10485760

// 1 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -20971520

// 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -41943040

// 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -83886080

// 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -167772160

// 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -335544320

// 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -671088640

// 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -1342177280

// 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is 1610612736

// 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 // 正数

// val is -1073741824

// 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

// val is -2147483648

val = -5;

for (int i = 0; i < 32; ++i)

{

for (long t = 2147483648; t > 0; t = t / 2)

{

if ((val & t) != 0)

{

Console.Write("1 ");

}

if ((val & t) == 0)

{

Console.Write("0 ");

}

}

Console.WriteLine();

Console.WriteLine("val is {0}", val);

val = val << 1;

}

}

}

}