Битовые операции (для кодирования/декодирования/шифрования)

Дополнение ~

Побитовая конъюнкция &

1 число 1101

2 число 1011

1101 & 1011 == 1001 (побитовая) == 1 & 1 == 1, 1 & 0 == 0, 0 & 1 == 0, 1 & 1 == 1

Побитовая дизъюнкция |

00000011 (2) == 3 (10)

~11111100 == 00000011

Сдвиги (**загугли**)

1. Сдвиг влево. <<. Справа всегда 0 дописывается  
   (0000100) << 1 == 0001000 (умножение на степень двойки)
2. Сдвиг вправо. >>. (если отрицательное, слева допишет 1, если положительное – 0)  
   (0001000) >> 1 == 0000100  
   (1001000) >> 1 == 1100100
3. Беззнаковый сдвиг вправо. >>>. Не учитывает знак числа, всегда 0 пишет.

Integer.ToBinaryString

Циклы:

while (логическое выражение) {операторы;} – пока выражение верно, повторяет операторы внутри блока

do {операторы;} while (логическое выражение)

Префиксная запись чуть быстрее

--b быстрее b--

массивы

Объявление

int[] m;

Тип[] имяПеременной

Определение (дать память под массив)

имяПеременной = new Тип[длинаМассива];

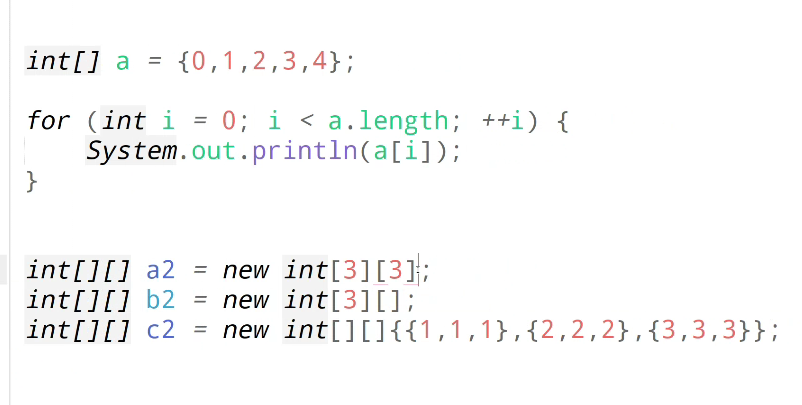
a = new int[5];

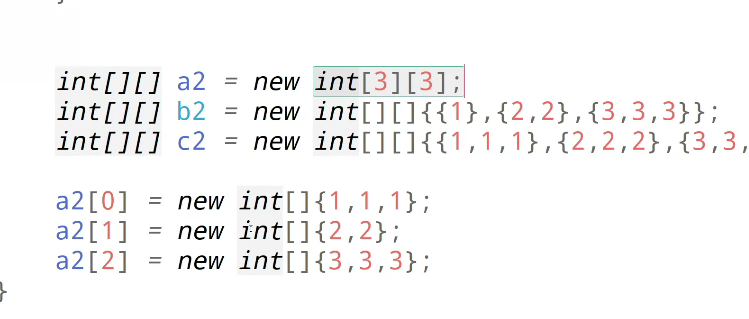
Обращение к элементам.

имяПеременной[индекс] = значение;

Элементы массива хранятся в памяти друг за другом. Массив дает самый быстрый доступ к элементу.

Двумерный массив – массив массивов

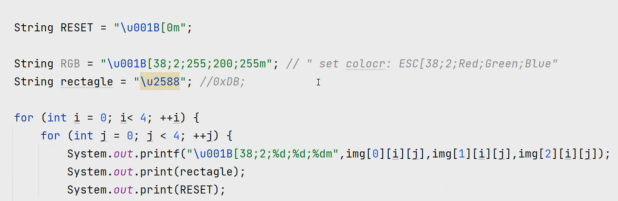


Можно создать массив в 2 элемента (в двухмерном массиве), даже если указали 3 элемента

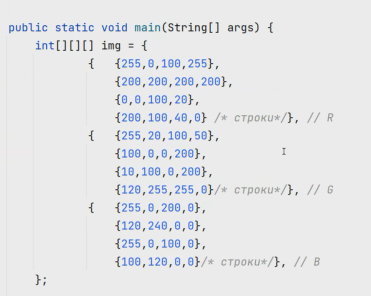
for (список выражений 1; логическое выражение; списокВыражений2) {операторы;}

Без вложенных функций решить дз в телеге

Нарисовать цветной прямоугольник







 - прямоугольник