Задание

Задайте значение y, например, с помощью потока ввода (или для отладки с помощью числового литерала в двоичном или 16-ричном коде)

С помощью побитовых операторов и операторов сдвига: выведите значение "y" на консоль в двоичном виде, например:
 если y==9, то
0 0 0 0 ... 0 0 1 0 0 1 (количество двоичных цифр зависит от платформы)

Подсказки

1) Биты должны выводиться слева направо, т.е. начиная со старшего разряда.
2) Количество битов числа (типа int) можно вычислить, но для простоты можно принять равным 32.
3) Имеет смысл сформировать маску mask, которая содержит 1 только в одном бите и использовать эту маску для получения значения этого бита в числе value
4) Можно задать начальное значение mask= 1<<31 (1 в старшем разряде)
4) С помощью операции value & mask можно определить значение одного бита.
5) Для получения следующей двоичной цифры надо изменить значение маски с помощью операции сдвига.

Задание

Введите с помощью потока ввода переменную типа int. Все биты, равные 1 сдвинуть вправо (влево).

1 способ

Формируем значение

Задаем маски:

```
int mask1 = 1, mask2 = 1;
// mask1  -  для проверки значения бита
// mask2  -  для установки нового значения бита

Цикл до  тех пор, пока все 32 бита не проверим
{
    Проверяем значение текущего бита числа (с помощью маски mask1):
    если бит==1, то
            {
            снимаем 1 в текущем бите исходного числа
            ставим в исходном числе 1 в  бите, заданном маской mask2
            сдвигаем маску установки бита (mask2) на 1 бит влево
            }
    сдвигаем маску проверки бита (mask1) на 1 бит влево
}
```

Формируем значение

Задаем маски: на самый “левый” бит и на самый “правый” бит: **2 способ**

unsigned int maskLeft = 0x80000000, maskRight = 0x01;

Цикл до тех пор, пока (maskLeft > maskRight)

{

Проверяем значение левого бита числа:

если “левый” бит==1, то

Проверяем значение правого бита числа:

если “правый” бит==0, то

1) снимаем 1 в левом бите

2) устанавливаем 1 в правом бите

3) сдвигаем maskLeft на 1 вправо

4) сдвигаем maskRight на 1 влево

иначе

сдвигаем maskRight на 1 влево

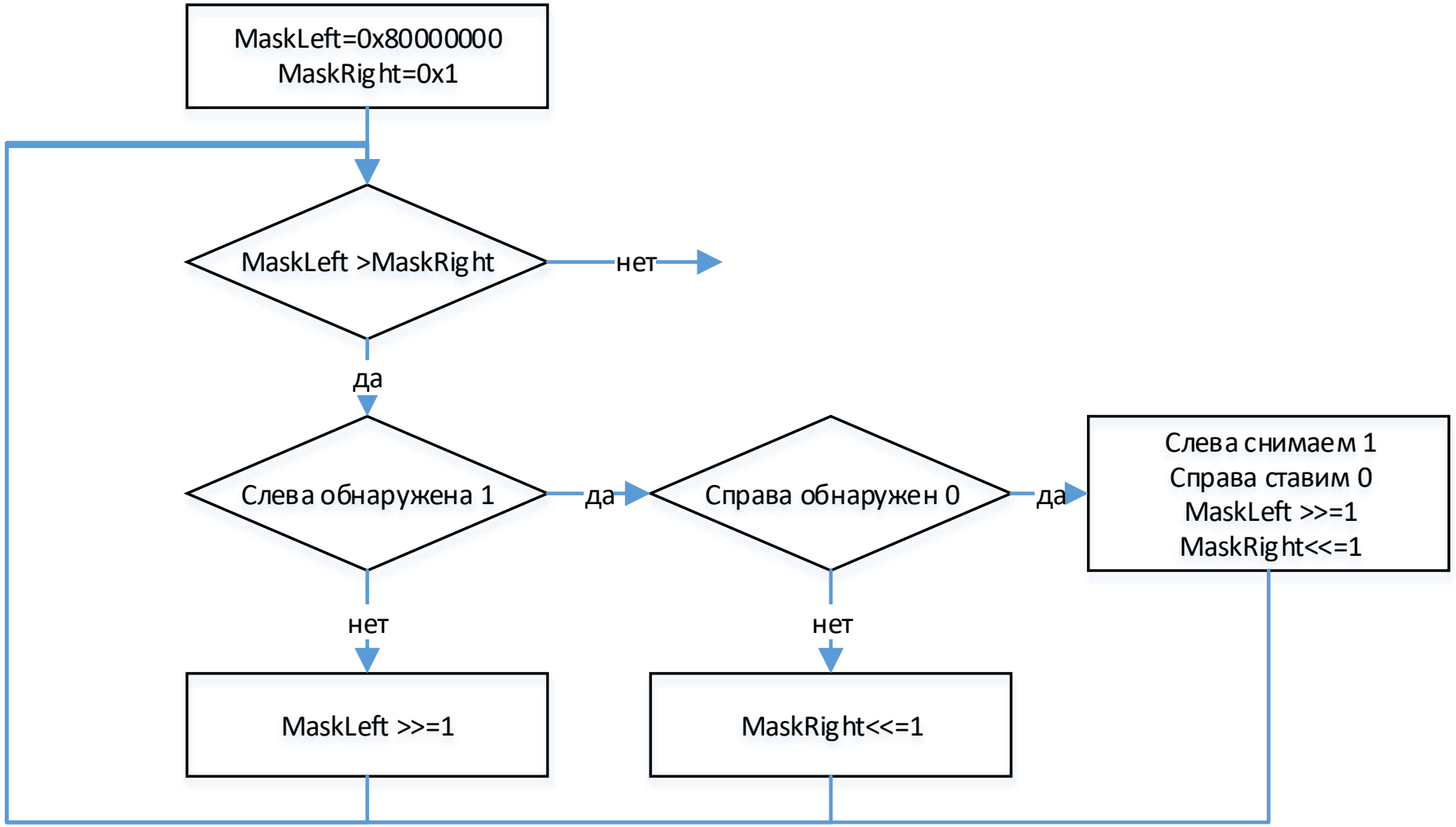
иначе

сдвигаем maskLeft на 1 вправо

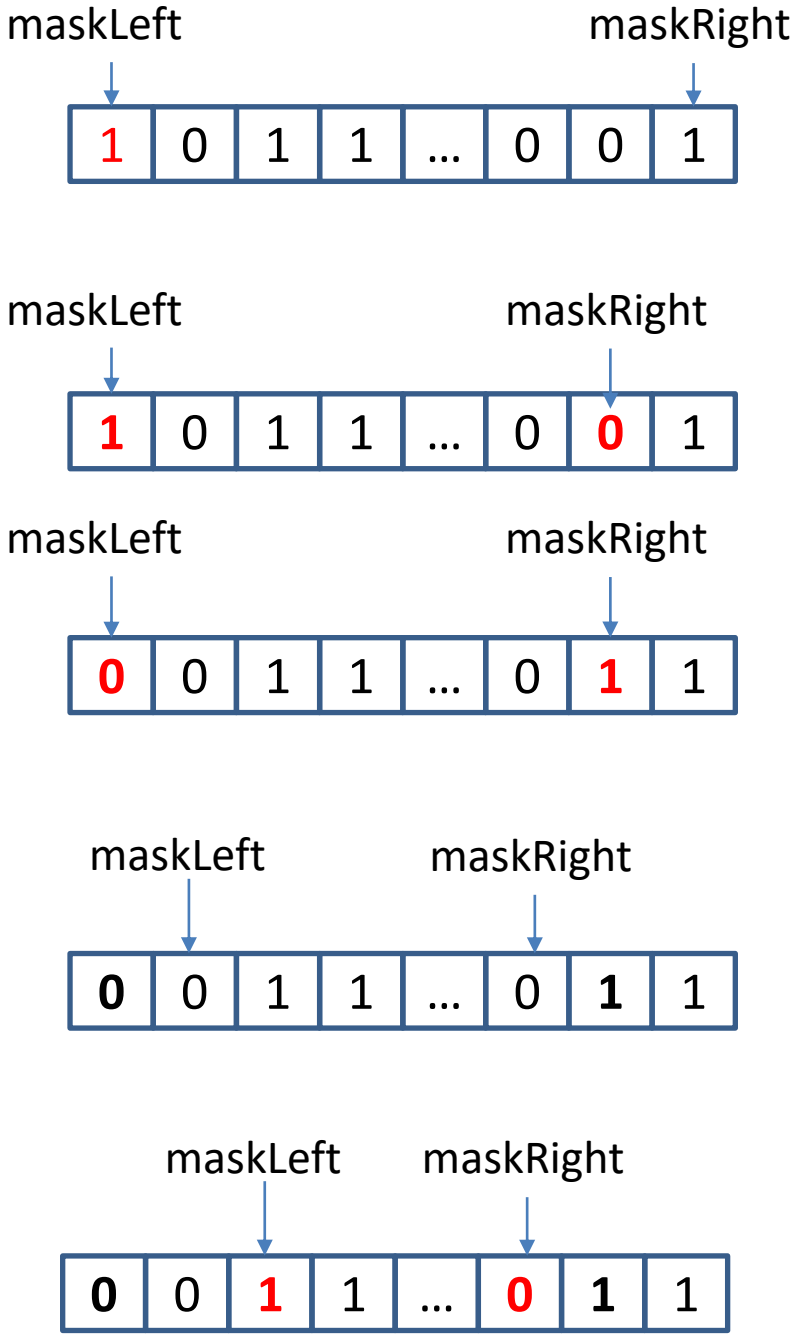
}

Задание

2 способ



2 способ


maskLeft
maskRight
1 0 1 1 … 0 0 1
maskLeft
maskRight
1 0 1 1 … 0 0 1
maskLeft
maskRight
0 0 1 1 … 0 1 1
maskLeft
maskRight
0 0 1 1 … 0 1 1
maskLeft
maskRight
0 0 1 1 … 0 1 1

3 способ

Формируем значение value

Задаем маски: на самый “левый” бит и на самый “правый” бит:

unsigned int maskLeft = 0x80000000, maskRight = 0x01;

Цикл до тех пор, пока (maskLeft > maskRight)

{

Выполняем цикл до тех пор, пока (maskLeft ==0)

сдвигаем maskLeft на 1 вправо

Выполняем цикл до тех пор, пока (maskRight ==1)

сдвигаем maskRight на 1 влево

Если (maskLeft <= maskRight),

то покидаем цикл

1) снимаем 1 в левом бите

2) устанавливаем 1 в правом бите

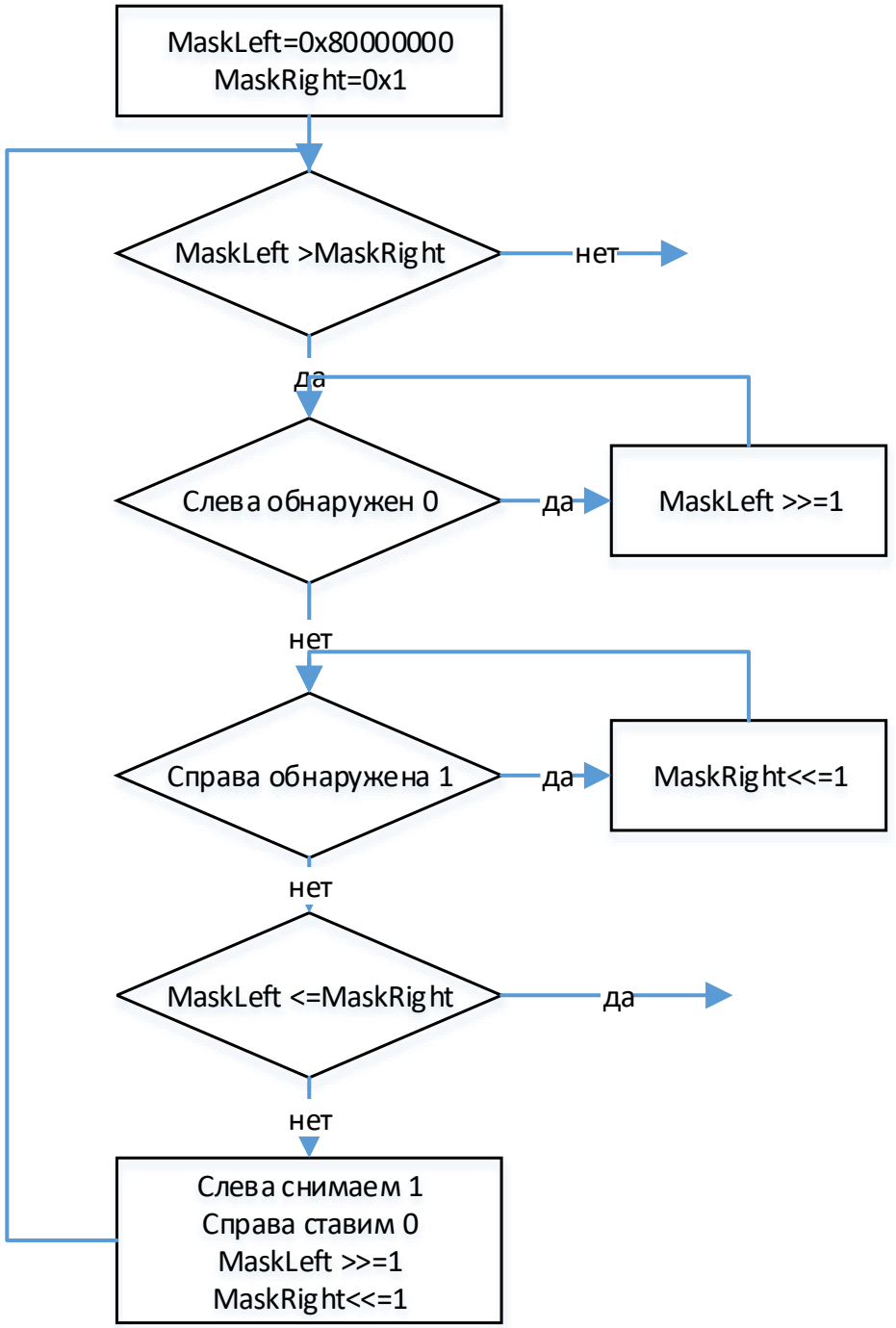
3) сдвигаем maskLeft на 1 вправо

4) сдвигаем maskRight на 1 влево

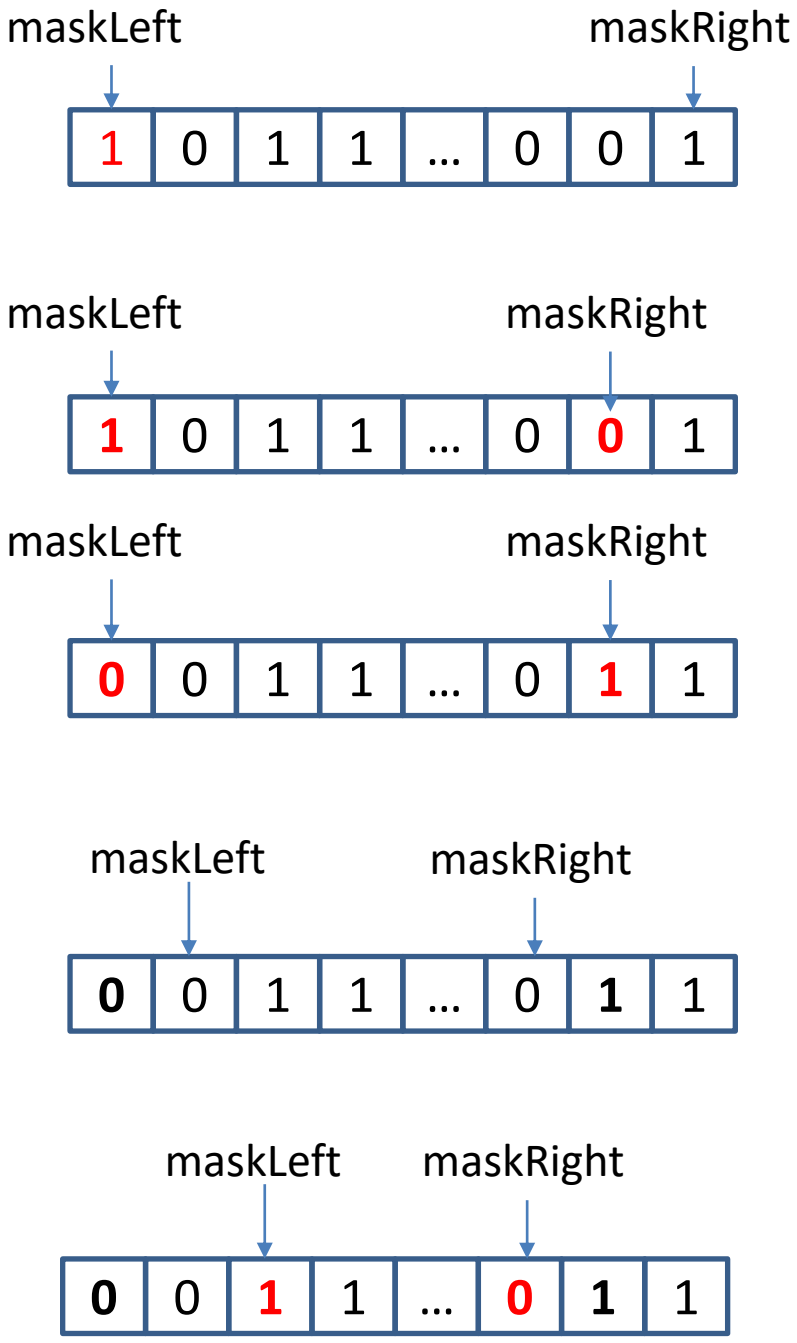
}

Задание

3 способ



3 способ


maskLeft
maskRight
1 0 1 1 … 0 0 1
maskLeft
maskRight
1 0 1 1 … 0 0 1
maskLeft
maskRight
0 0 1 1 … 0 1 1
maskLeft
maskRight
0 0 1 1 … 0 1 1
maskLeft
maskRight
0 0 1 1 … 0 1 1