Язык JavaScript (основы)
Дополнительный материал

ПОБИТОВЫЕ ОПЕРАТОРЫ

Побитовые операторы рассматривают аргументы как 32-разрядные целые числа и работают на уровне их внутреннего двоичного представления.

Поддерживаются следующие побитовые операторы:
- AND(и) (&)
- OR(или) (|)
- XOR(побитовое исключающее или) (^)
- NOT(не) (~)
- LEFT SHIFT(левый сдвиг) (<<)
- RIGHT SHIFT(правый сдвиг) (>>)
- ZERO-FILL RIGHT SHIFT(правый сдвиг с заполнением нулями) (>>>)

ПОБИТОВЫЕ ОПЕРАТОРЫ

Логические операции И, ИЛИ, исключающее ИЛИ и НЕ могут быть описаны с помощью таблиц истинности:

Логический оператор И

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

Логический оператор ИЛИ

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

Логический оператор исключающее ИЛИ

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

Логический оператор НЕ

X	NOT X
0	1
1	0

ПОБИТОВЫЕ ОПЕРАТОРЫ

В побитовых операциях значение бита, равное 1, рассматривается как логическая истина, а 0 как ложь. Побитовое И (оператор &) берёт два числа и логически умножает соответствующие биты. Например:

```
var a = 3;                  00000011
var b = 8;                  00001000
var c = a & b;              ↓↓↓↓↓↓↓↓
console.log(c); // 0        00000000

var a = 31;                 00011111
var b = 17;                 00010001
var c = a & b;              ↓↓↓↓↓↓↓↓
console.log(c); // 17       00010001
```

Логический оператор И

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

ПОБИТОВЫЕ ОПЕРАТОРЫ

Аналогично работает операция побитового ИЛИ (оператор |), за исключением того, что она логически суммирует соответствующие биты чисел без переноса. Например:

```
var a = 15;                00001111
var b = 11;                00001011
var c = a | b;             ↓↓↓↓↓↓↓↓
console.log(c); // 15      00001111

var a = 33;                00100001
var b = 11;                00001011
var c = a | b;             ↓↓↓↓↓↓↓↓
console.log(c); // 43      00101011
```

Логический оператор ИЛИ

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

ПОБИТОВЫЕ ОПЕРАТОРЫ

Побитовое отрицание (оператор ~) работает не для отдельного бита, а для всего числа целиком. Оператор инверсии меняет ложь на истину, а истину на ложь, для каждого бита. Например:

```
var a = 65;
var b = ~a;
console.log(b);// -66
```

```
01000001
↓↓↓↓↓↓↓↓
10111110
```

Логический оператор НЕ

X	NOT X
0	1
1	0

ПОБИТОВЫЕ ОПЕРАТОРЫ

Исключающее ИЛИ (оператор ^) применяет побитовую операцию XOR. Например:

```
var a = 12;                00001100
var b = 85;                01010101
var c = a ^ b;             ↓↓↓↓↓↓↓↓
console.log(c); // 89      01011001
```

Логический оператор исключающее ИЛИ

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

!!! Обратимая операция.

ПОБИТОВЫЕ ОПЕРАТОРЫ

Операций сдвига: битовый сдвиг влево (оператор <<) и битовый сдвиг вправо (оператор >>). Битовый сдвиг вправо сдвигает биты числа вправо, слева добавляется *копия* крайнего-левого бита. Битовый сдвиг влево: сдвигает биты влево, дописывая справа нули. Вышедшие за пределы числа биты отбрасываются. Например, сдвиг числа 5 влево на 2 позиции

$$00000101 << 2 == 00010100$$

Сдвиг числа 19 вправо на 3 позиции

$$00010011 >> 3 == 00000010$$

Так как сдвиг вправо (>>) дописывает слева нули, то для целых чисел операция равносильна целочисленному делению пополам, а сдвиг влево умножению на 2.

Шифрование строки по ключу.

Пользователь вводит произвольную строку. Затем вводит ключ (один символ).

Затем программа шифрует текст, а именно применяет операцию исключающего ИЛИ для каждого символа введённой строки с использованием ключа.

Вывести зашифрованное сообщение.