また２進数について

（シェルで、node で実行可能、.exitで退出）

- x(数値) を n進数に変換: (x).toString(n)
- AND: &
- OR: |

```
> (1234).toString(2)
'10011010010'
> (4321).toString(2)
'1000011100001'
> 1234 & 4321
192
> (192).toString(2)
'11000000'
> (1234 & 4321).toString(2)
'11000000'
> 1234 | 4321
5363
> (1234 | 4321).toString(2)
'1010011110011'
```

- n進数y(文字列)を Int に変換: parseInt(y, n)

```
> (192).toString(16)
'c0'
> parseInt('c0', 16)
192
```

- XOR: ^
- LEFT SHIFT: <<
- RIGHT SHIFT: >>

```
> 1234 ^ 4321
5171
> (1234 ^ 4321).toString(2)
'1010000110011'
> 1234 ^ 4321 ^ 4321
1234
> 1234 ^ 4321 ^ 1234
4321
```

```
> (1234).toString(2)
'10011010010'
> 1234 << 1
2468
> (1234 << 1).toString(2)
'100110100100'
> 1234 << 2
4936
> (1234 << 2).toString(2)
'1001101001000'
```

```
> 1234 >> 1
617
> (1234 >> 1).toString(2)
'1001101001'
> 1234 >> 2
308
> (1234 >> 2).toString(2)
'100110100'
```

プログラミングに利用する数学

- 整数
  - 素数(Prime Number)と合成数(Composite Number)
- 小数点数
  - 固定小数点数 (fixed-point number)
  - 浮動小数点数 (floating-point number) 参考
- 演算子 (Javascript 例)
  - 算術演算子(+, −, *, /, **)
  - 関係演算子(<, >, <=, >=)
  - 等値演算子(==, !=, ===, !==)
- 優先順位 (Javascript 例)
- 関数/写像 (Function/Mapping)
- ベクトル (Vector) 参考
- 行列 (Matrix) 参考

正規表現（特定なパターンを表現する）

- 用途
  - 高度な検索及び置換
    - 表記統一 (例：¥5678 → 5678 円)
  - パターンのマッチングチェック
    - markdown のタイトル行かどうかの判断 (例：^#+\s)
- 正規表現学習参考資料例
  - サルにもわかる正規表現入門
  - Javascript ドキュメント