東京工科大学 講義資料

「データ構造とアルゴリズムの基礎」
第4回　木構造と再帰呼び出し

2017年10月20日

柴田 博仁
hshibata@edu.teu.ac.jp

感想に対するフォードバック (1)

- 私は図書館司書の資格を取得しようと考えており、図書カードの整理の仕方を考える問いがあった際にとても考えさせられました。また、図書委員をしていたことがありましたので、図書カードの整理にはいつも苦労していた記憶があります。
- 分かりやすいように学生たちに身近な例で例える事が多く、分かりやすく伝えたいんだ、という意志がよく伝わってきました。おかげで割とすんなり理解できたような気がします。これからも良い例をお願いします。
- データ構造に関しての、配列とリストの違いがとても勉強になりました。似たような意味を成すプログラムでも、データを追加・挿入する際の計算量が大きく違い、それぞれにメリット・デメリットが存在するという事がとても分かりやすかったです。それらの特徴をわきまえて、今後のコーディングに活かしていきたいと思います。

感想に対するフォードバック (2)

- 前のほうに座ると理解力が上がるという話は私自身の経験からみて実際にそう思う。高校生の時に目が悪くなってから前のほうに座るようになってからそれまでよりも、一度に理解できることが多くなった。効率的に勉強をするのなら前に座るのは必須だと考える。
- FIFO、LIFOは日常に溢れているということを再認識しました。
- 今回の講義で1番印象に残った話が算術と面積の話です。2桁×2桁の計算は暗算でしていましたが、そういう考え方もあるのかと思い驚きました。その考えを出来る人は本当に頭がいいと感じました。

LIFO (スタック)、 FIFO (キュー) の例

FIFOの例

- 滝の水
- ホースの水：最初にホースに入った水が一番最初に出る。
- ペン式の消しゴムの消しゴム：一番最初にペンに入れた消しゴムが一番最初に出る。
- 入口と出口が別のエレベーター

LIFOの例

- 満員電車
- 海の波の行って返ってくる様子
- トラックの積み荷
- タクシーの運転手:タクシーに一番最初に乗り、仕事終わりに降りる
- ガムテープ
- お団子
- キーボードの入力と削除

LIFO/FIFOの対比

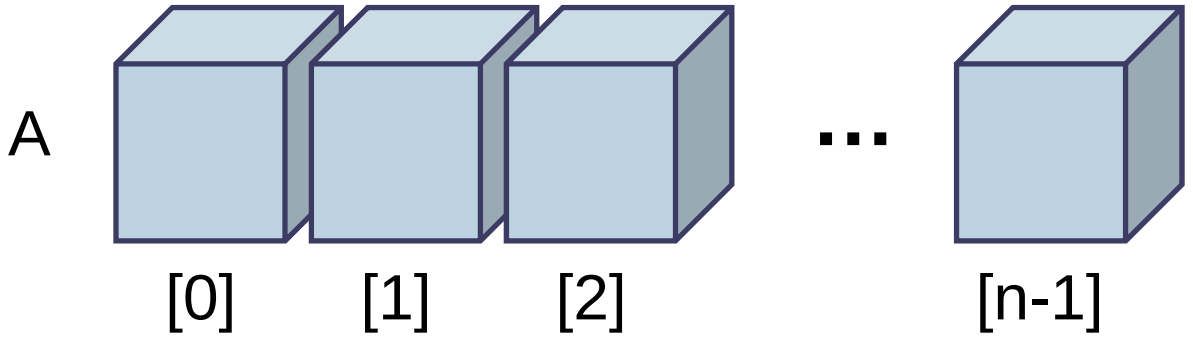
- 病院に行く際に予約していかない人はLIFO、予約してから来る人はFIFO。
- スーパーでお店の人の承認の並べ方はFIFO、お客の商品の取り方はLIFO。
- よい学生はFIFO(きちんと開始前に座り、終了後は次の授業へ向かう)で、悪い学生はLIFO(授業に遅刻してきて、授業が終わっても寝ていて気づかない)

前回の復習

配列 (1) 配列とは

番号付きのコインロッカーのようなもの

- 箱を動かすことはできない
- 箱を増やすことはできない
- 番号を指定してのアクセスが高速

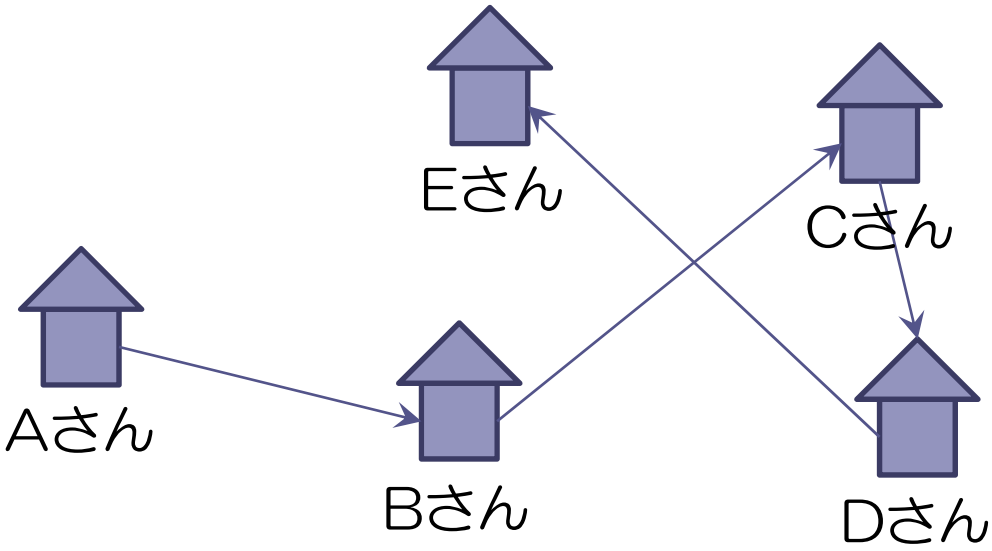


- 配列名：A
- 配列のサイズ：n
- 配列のインデックス：0 … n-1

連結リスト (1) 連結リストとは

電話を使った連絡網のようなもの

- 私は自分が誰かを知っている
- 自分の後に誰に電話するかを知っている
  - ✓でも、誰からかかってくるか知らない
  - ✓自分が何番目かはわからない
  - ✓全体で何人いるのかも知らない



木構造

いろいろなデータ構造

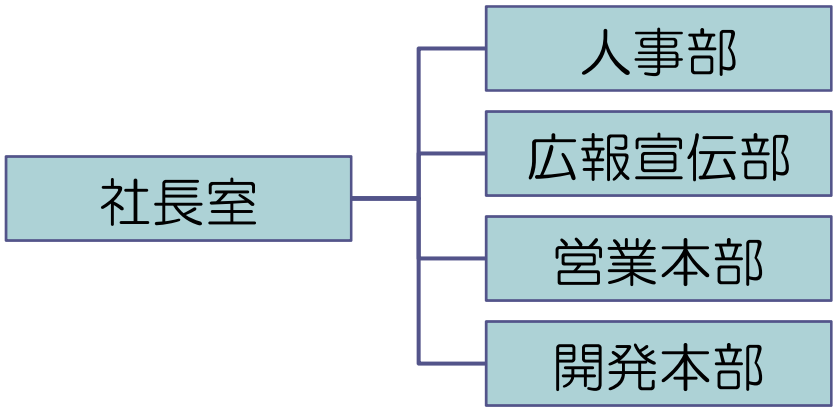
- 配列
- 連結リスト
- スタック
- キュー
- 木 (ツリー)
- ハッシュ
- 行列

木構造の例: 大学の組織図

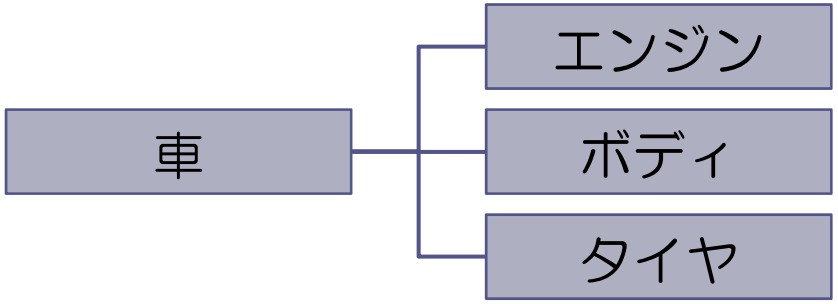
さまざまな木構造

ものごとは、細分化した方が、機能を分割したほうが考えやすいし、管理しやすい。さらに、トップは1つでないと、管理しにくい、責任の所在が不明になる。

機能による分割
会社の組織図

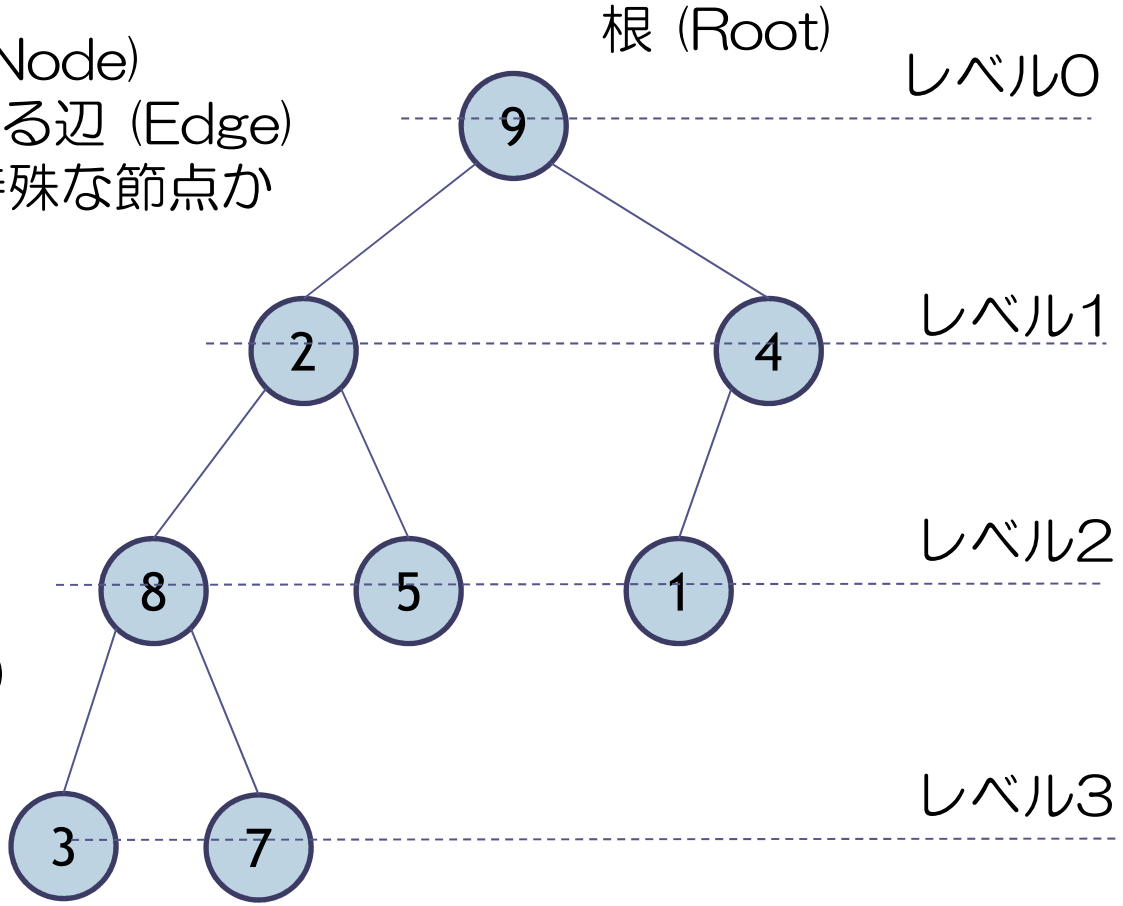


部位による分割
車の構成要素

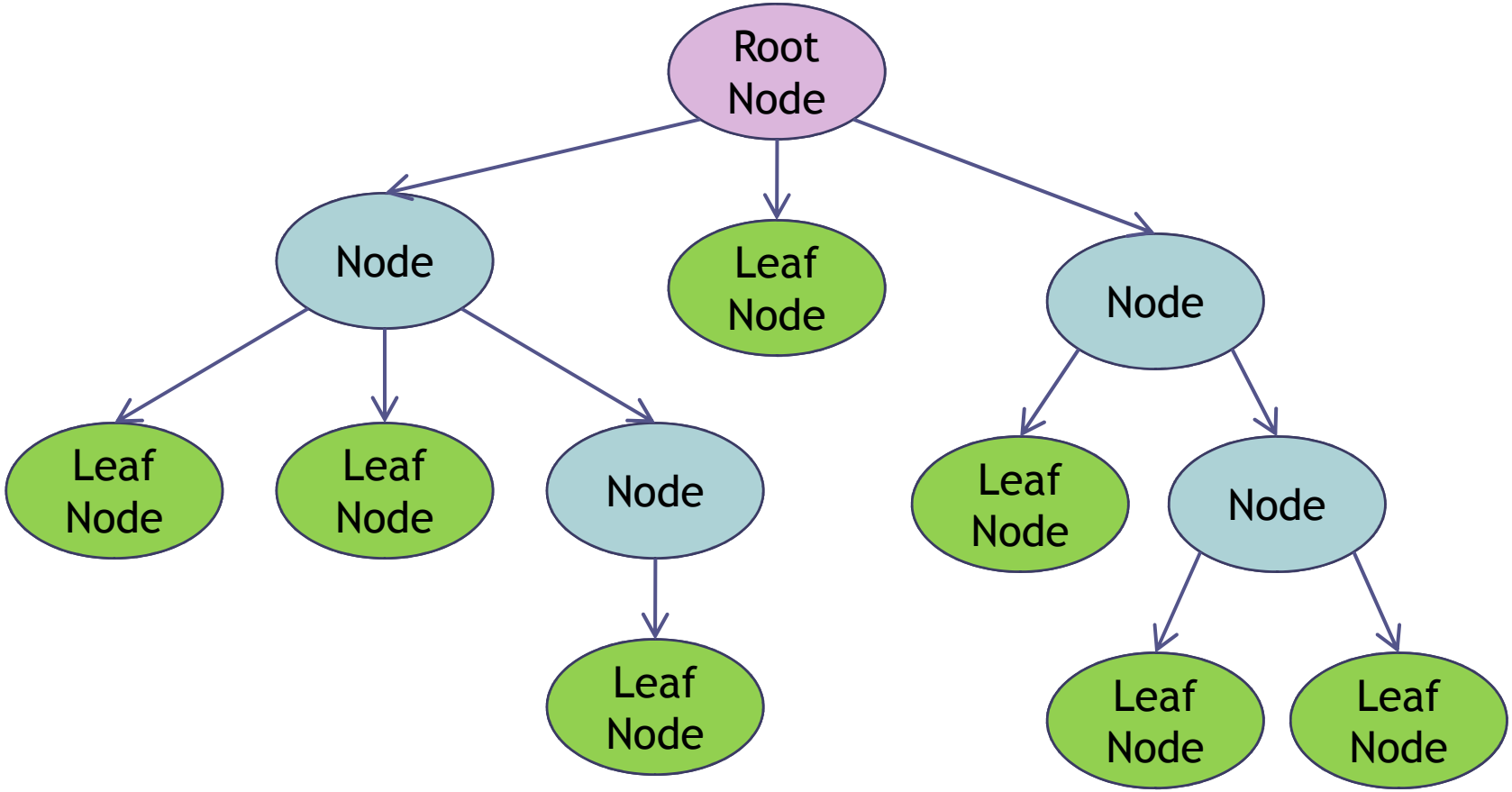


木構造の表現

円で表現する節点 (Node)
円を結ぶ線で表現する辺 (Edge)
根 (Root) という特殊な節点から次々と分岐
ループはない

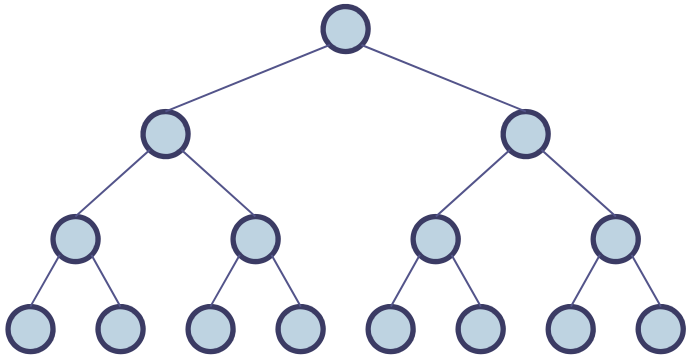


木構造の表現(2)


Root Node
Node
Leaf Node
Node
Leaf Node
Leaf Node
Node
Leaf Node
Node
Leaf Node
Leaf Node
Leaf Node

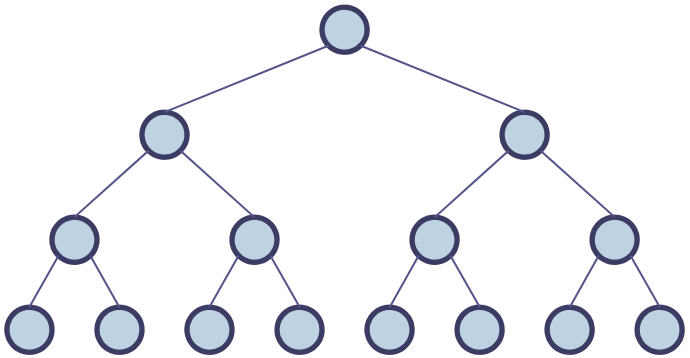
2分木と完全2分木

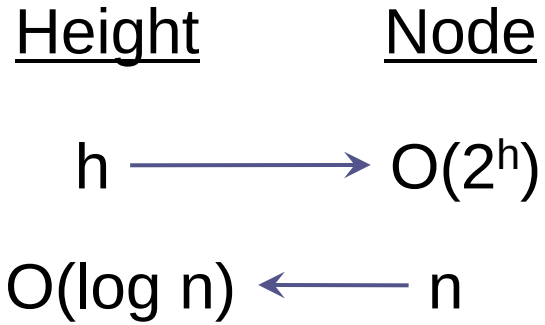
- 2分木
  - ✓木のどの節点も2個以下の子しかもたない (各節点での枝の分岐数が2まで)
- 完全2分木
  - ✓全ての葉が同じレベルであり、葉以外の全ての節点が2つの子をもつ



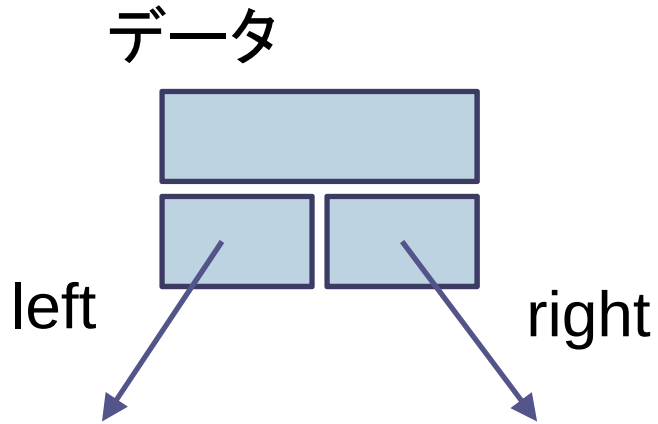
完全2分木の性質

- 高さ h の完全2分木の節点の数は
  $1+2+4+\ldots+2^{h-1} = 2^h - 1 = O(2^h)$
- 節点の数が n の完全2分木の高さは
  $\log (n+1) = O(\log n)$





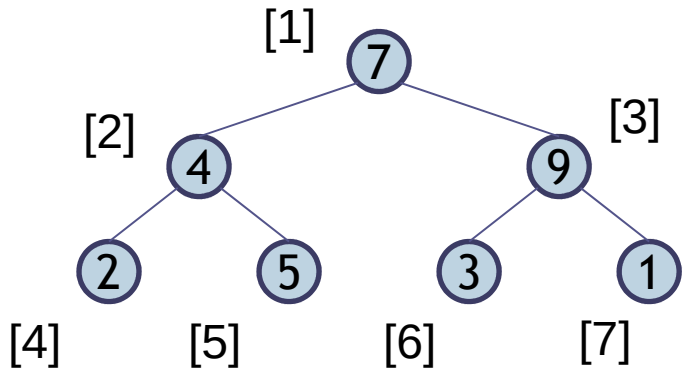
2分木の表現 (節点をオブジェクト化)

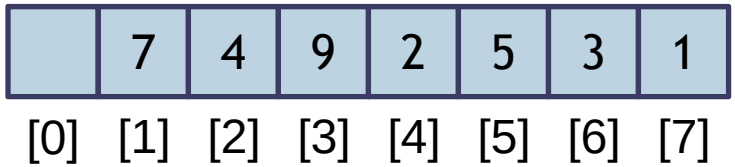



```
class Node {
        int data;          // データを格納
        Node left;         // 一方の子ノードへのポインタ
        Node right;        // 他方の子ノードへのポインタ
}
```

2分木の表現 (配列を利用)

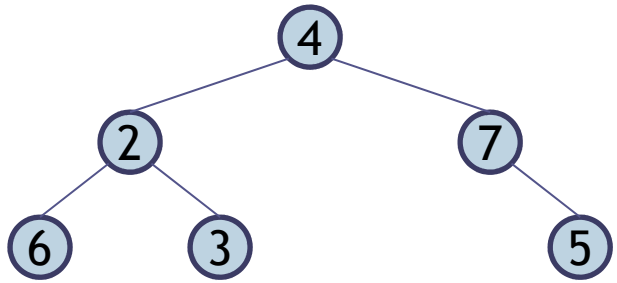
1. ルートのインデックスを1とする
2. インデックス i のノードの2つの子について、
   - ✓ 左の子のインデックスは 2i
   - ✓ 右の子のインデックスは 2i+1

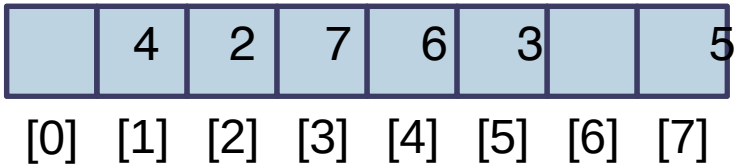




考えてみよう

下の木構造を配列で表現してみよう





再帰呼び出し

再帰

「再帰呼び出し」

- 関数の中でその関数自身を呼び出すこと

「再帰アルゴリズム」

- 再帰的な関数を含むアルゴリズム

再帰呼び出しすべき問題例

問題
ある細胞は、1分経過すると分裂し、数が2倍になる。
しかし、分裂直後に全細胞のうち4個が死滅する。
最初に試験管に10個の細胞を入れる。
n分後の細胞の数は何個か？

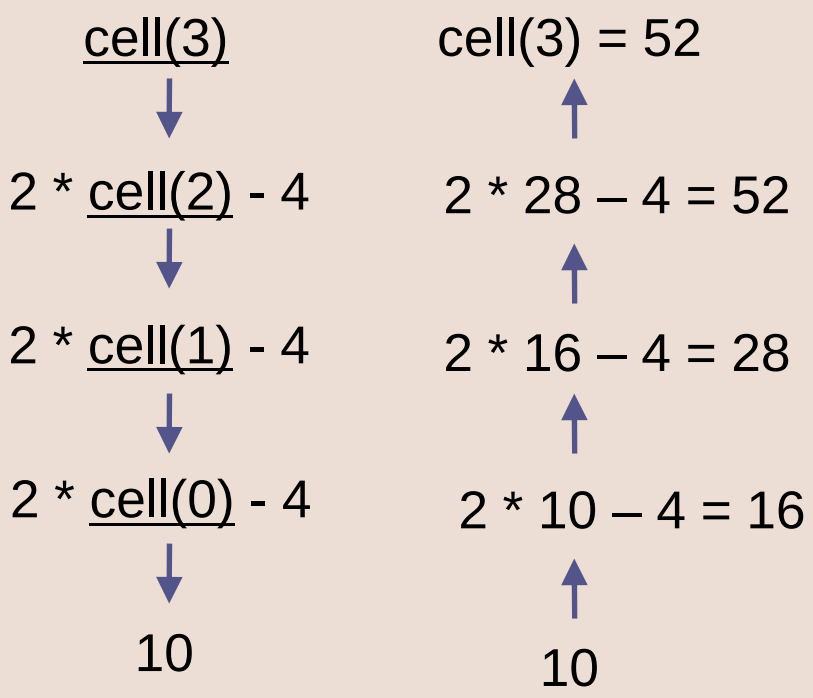
n分後の細胞の数を c(n) とする。
c(n) を n の関数として表現することは難しい。
しかし、c(n) と c(n−1) の関係を記述することは容易。
c(n) = 2 * c(n−1) − 4
また、初期状態は c(0) = 10

数列の漸化式のよう

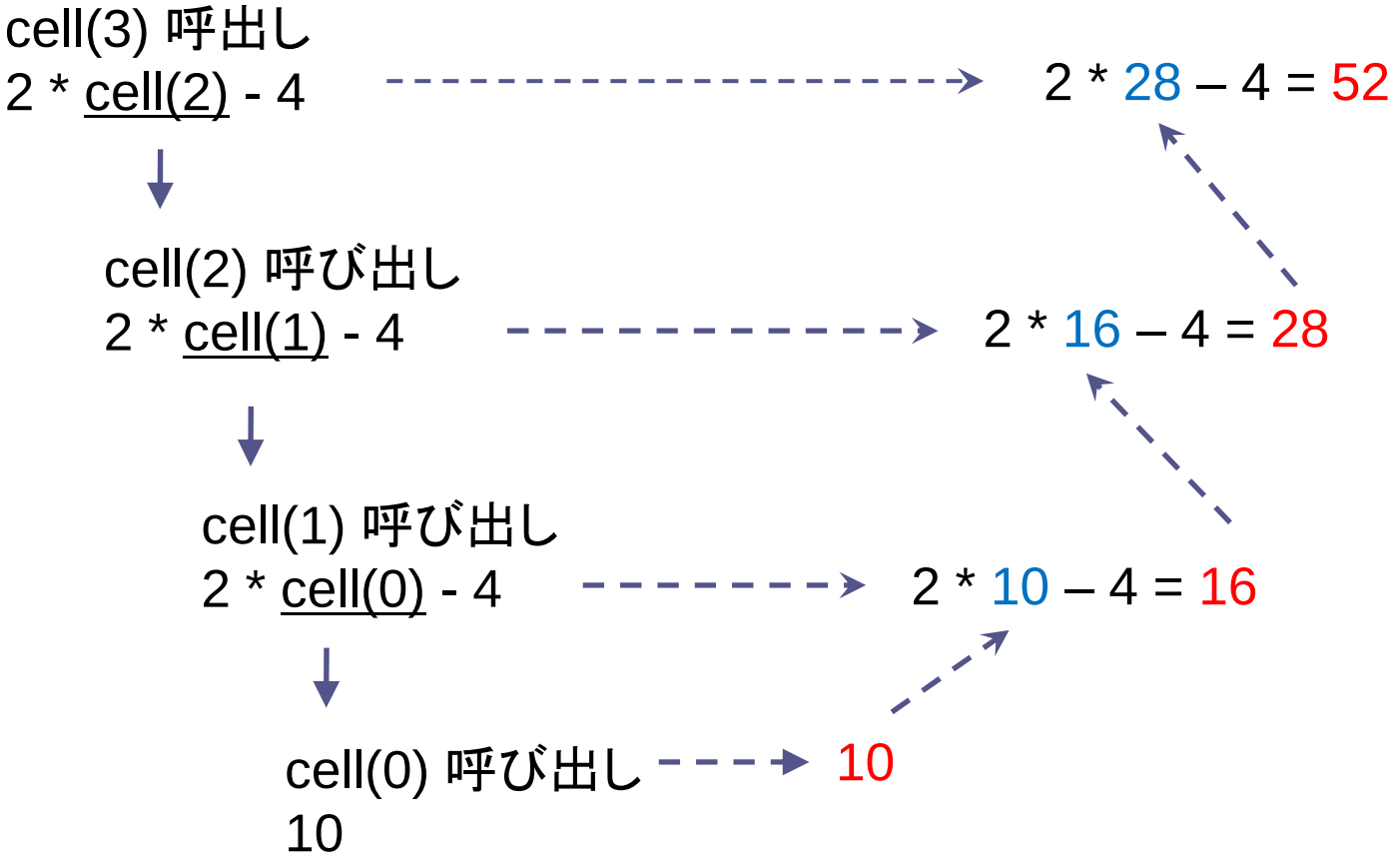
再帰呼び出しの例


```
// n分後の細胞の数を返す関数
int cell(int n)
{
    if (n == 0) {
        // 初期の細胞数
        return 10;
    } else {
        // 再帰的な呼び出し
        return 2 * cell(n-1) – 4;
    }
}
```

再帰呼び出しの図式化



呼び出しのタイミング (答えがわかる順番)



- ✓ すぐに結論が出せない。部下に事情を聴いてから (e.g. 社長によるトラブルの事情説明)
- ✓ すぐに納品できない。下請けに依頼してから (e.g. オーダーメイドの特注ドレス)

整数の和を求める再帰アルゴリズム

```
// n個の配列Aの和を求める
int sum1(int[] A)
{
        int sum = 0;
        for (int i = 0; i < n; i++)
                sum = sum + A[i];
        return sum;
}

// n個の整数の和を求める
int recursive_sum1(A[0], A[1], ..., A[n-1])
{
        if (引数がA[0]のみ) {
                return A[0];
        } else {
                return recursive_sum1(A[0], ..., A[n-2]) + A[n-1];
        }
}
```

どちらが効率的でしょか？

nが2のべき乗のとき、$a^n = a^{n/2} * a^{n/2}$ が成り立つ。
この性質を利用して、2つの再帰アルゴリズムを考える。

```
pow1(n)
{
    if (n == 1) {
        return a;
    } else {
        return pow1(n/2) * pow1(n/2);
    }
}
```

```
pow2(n)
{
    if (n == 1) {
        return a;
    } else {
        p = pow2(n/2);
        return p*p;
    }
}
```

練習: フィボナッチ数列

フィボナッチ数列とは、
F(0) = F(1) = 1
F(n) = F(n −1) + F(n − 2)

[問題1] フィボナッチ数列を求める再帰アルゴリズムを書いてみよう。
  int fib(int n)
[問題2] このアルゴリズムの n=4 の場合の再帰呼び出しを図式化してみよう。

おわりに

Java開発環境の確認

- NetBeansの設定
- NetBeansの使い方

課題1

次のような出力を行うプログラムを for 文と if 文を用いて作成せよ

出力結果

0) Hello World!
1) Hello World!
2) Hello World! (2回目だよ)
3) Hello World!
4) Hello World!
5) Hello World!
6) Hello World!
7) Hello World! (7回目だよ)
8) Hello World!
9) Hello World!

ヒント

```
int i = 5;
System.out.println(i + “) Hello”);
```

→

5) Hello

課題の提出では、プログラムをそのままコピーしてメールにはりつけよ(添付ファイルにはしないこと)

課題2 (必須ではない、おまけ)

フィボナッチ数列のプログラムを作成し、fib(30)の値を求めよ。

課題提出では、fib(30)の結果のみでよい

課題の提出

- 提出先： hshibata@edu.teu.ac.jp
- 期限：10/24(火) 22:00
- メールのタイトルは「ADS Ex04 CXXXXXX 柴田博仁」
  - ✓ 氏名以外は全て半角 (スペースも半角)

  ※ タイトルに間違いがある場合は「未提出」とみなすことも

  ※ メールは何度も送らないこと。複数ある場合は最初のメールを提出とみなす
- メールの本文に
  - ✓ [1] 本日の感想
  - ✓ [2] 課題1
  - ✓ [3] 課題2