

=====

## B\*-Tree の挿入アルゴリズム

=====

1. proc insert\_record(record)
2.   Input:
3.     record : 挿入するレコード
4.     entry = (key 値, record)
5.     insert (ルートノードのポインタ, entry)

1. proc insert (P,record)
2. Input:
3.   P : 対象ノードへのポインタ(最初は root)
4.   record : 挿入するレコード
5.   output
6.   newEnt : 親ノードに挿入するエントリ (なければ NULL)
7.   N = P のさすノード
8.   if N が non-leaf ノードである :
9.      $K_i < \text{entry の key 値} \leq K_{i+1}$  となる i を探す
10.    newEnt\_fromchild = insert( $P_i$ , entry) //再帰
11.    if N に空きがある :
12.     newEnt\_fromchild を挿入して return null;
13.    else : //N に空きがない場合
14.     newEnt = divide\_nonleaf(N)
15.     if ノード N が root ノードである :
16.       新しい root ノード R を作成
17.       insert\_entry(R, newEnt)
18.       return null
19.     else:
20.       return newEnt

```

21.  else : //N が leaf ノードである
22.      if N に空きがある
23.          insert_entry(N, record)
24.          return null
25.      else
26.          newEnt = divide_leaf(N)
27.          if ノード N が root ノードである :
28.              新しい root ノード R を作成
29.              add_entry(R, newEnt)
30.              return null
31.          else
32.              return newEnt

```

---

### B\*-Tree のノード分割アルゴリズム

---

```

proc divide_nonleaf ( N )
    input
        N : 分割する non-leaf ノード
    output
        newEnt : 親ノードに挿入するエントリ
    process:
        ノード M を作る
        N の d+2 番目から 2d+1 番目までの key 値を M へ移動
        N の d+2 番目から 2d+1 番目までのポインタを M へ移動
        N の子孫の一番右のリーフノードから M の子孫の一番左の
            リーフノードへ兄弟ポインタを貼る
        newEnt = <N の d+1 番目の key 値, M へのポインタ>
        N の d+1 番目の key 値を削除
        return newEnt;

```

```

proc divide_leaf ( N )
  input :
    N : 分割する leaf
  output
    newEnt : 親ノードに挿入するエントリ

  process :
    ノード M を作る
    N の d+1 番目から 2d+1 番目までのエントリを M へ移動
    N から M へ兄弟ポインタを張る
    newEnt = <M の 1 番目の key 値, M へのポインタ>
    return entEnt;

```

=====

### B\*-Tree のエントリ追加アルゴリズム (non-leaf)

=====

```

insert_entry (newEnt)
  input:
    newEnt : 挿入するエントリ。key 値とポインタを持つ。
  process:
     $K_i \leq \text{newEnt の key 値} < K_{i+1}$  となる  $i$  を探す
     $i$  番目と  $i+1$  番目の間に空のエントリを追加する
      ( $i+1$  番目以降は 1 つずつ順番がずれる)
     $i+1$  番目のポインタに  $i+2$  番目のポインタを代入する
     $i+2$  番目のポインタに newEnt のポインタを代入する
     $i+1$  番目の key 値に newEnt の key 値を代入する

```