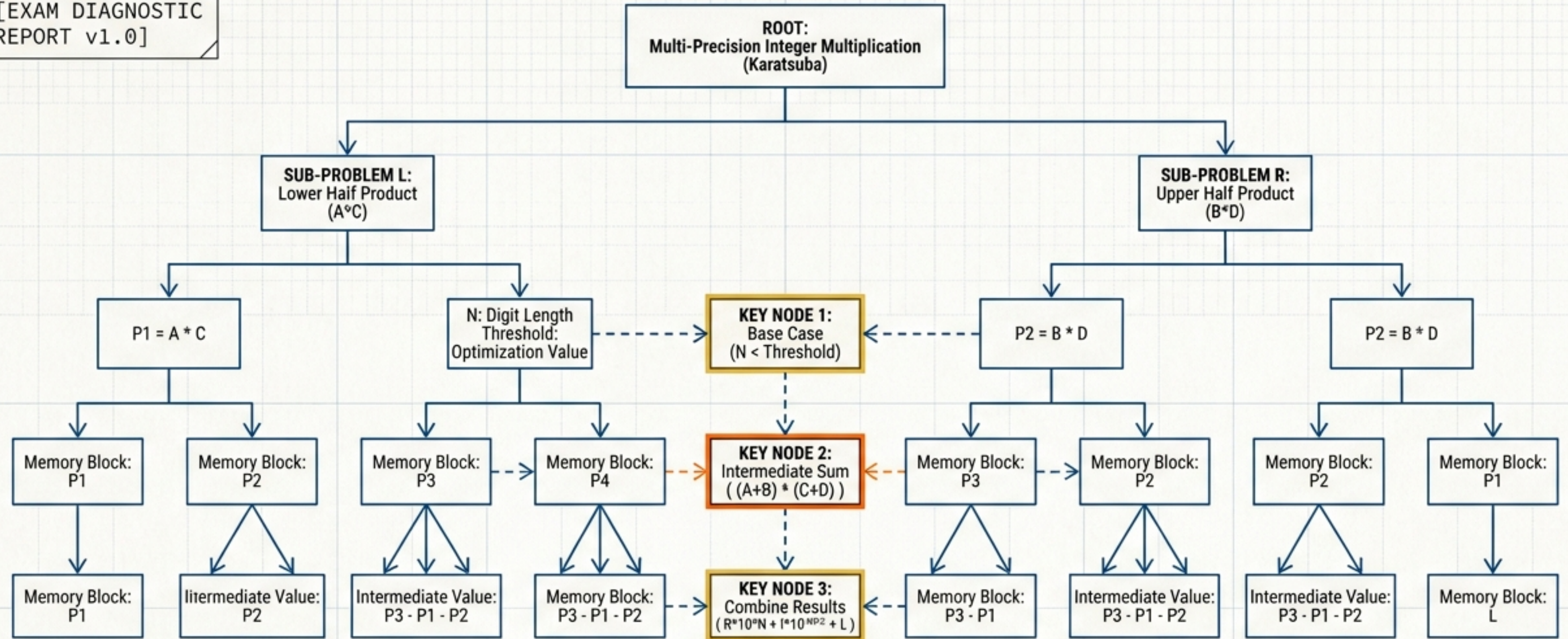


[EXAM DIAGNOSTIC
REPORT v1.0]



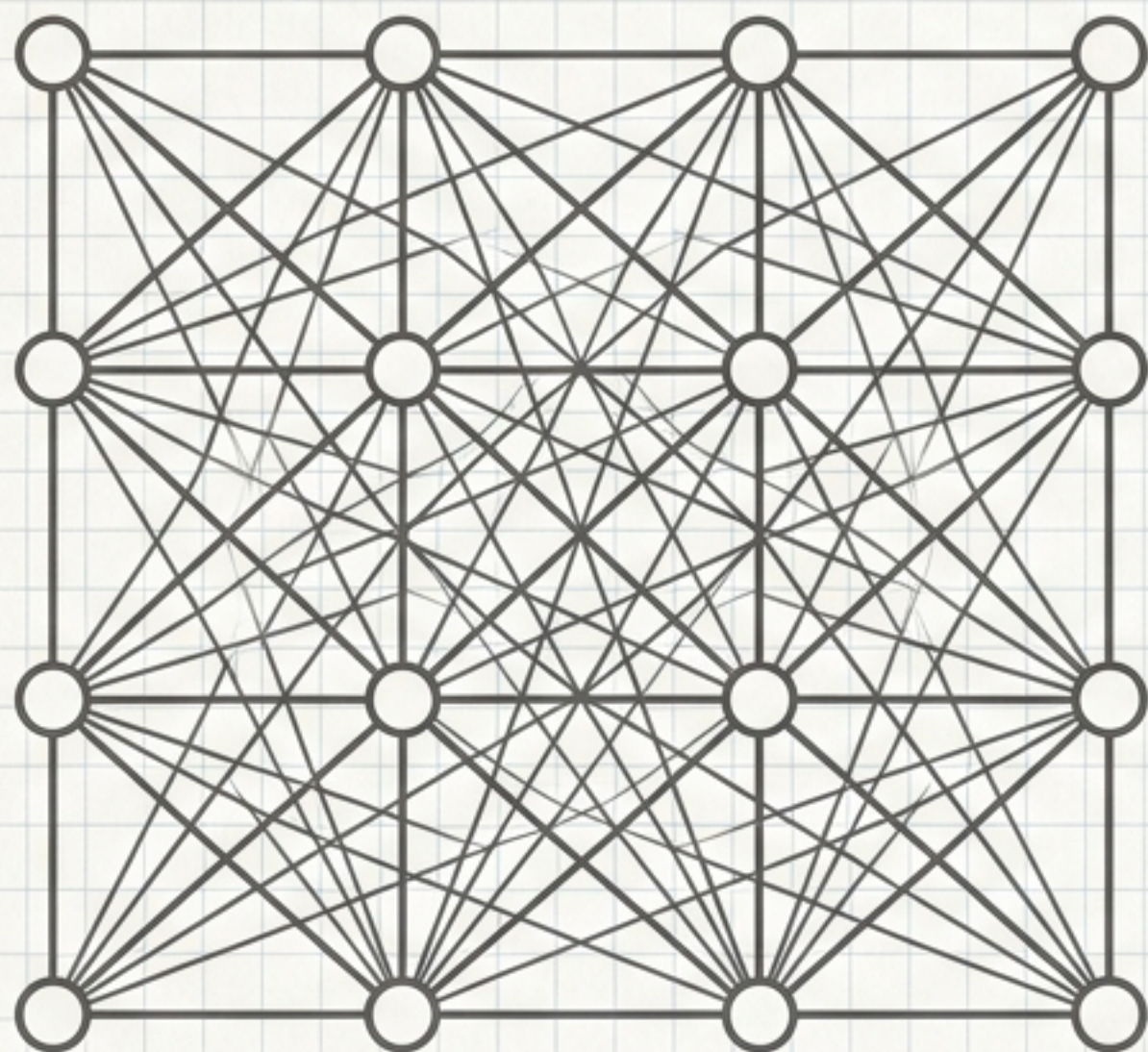
2023年春期 応用情報技術者試験 午後問 3 完全攻略ガイド

多倍長整数の演算 (カラツバ法) — 解答を導くための「思考のプロセス」とヒント集

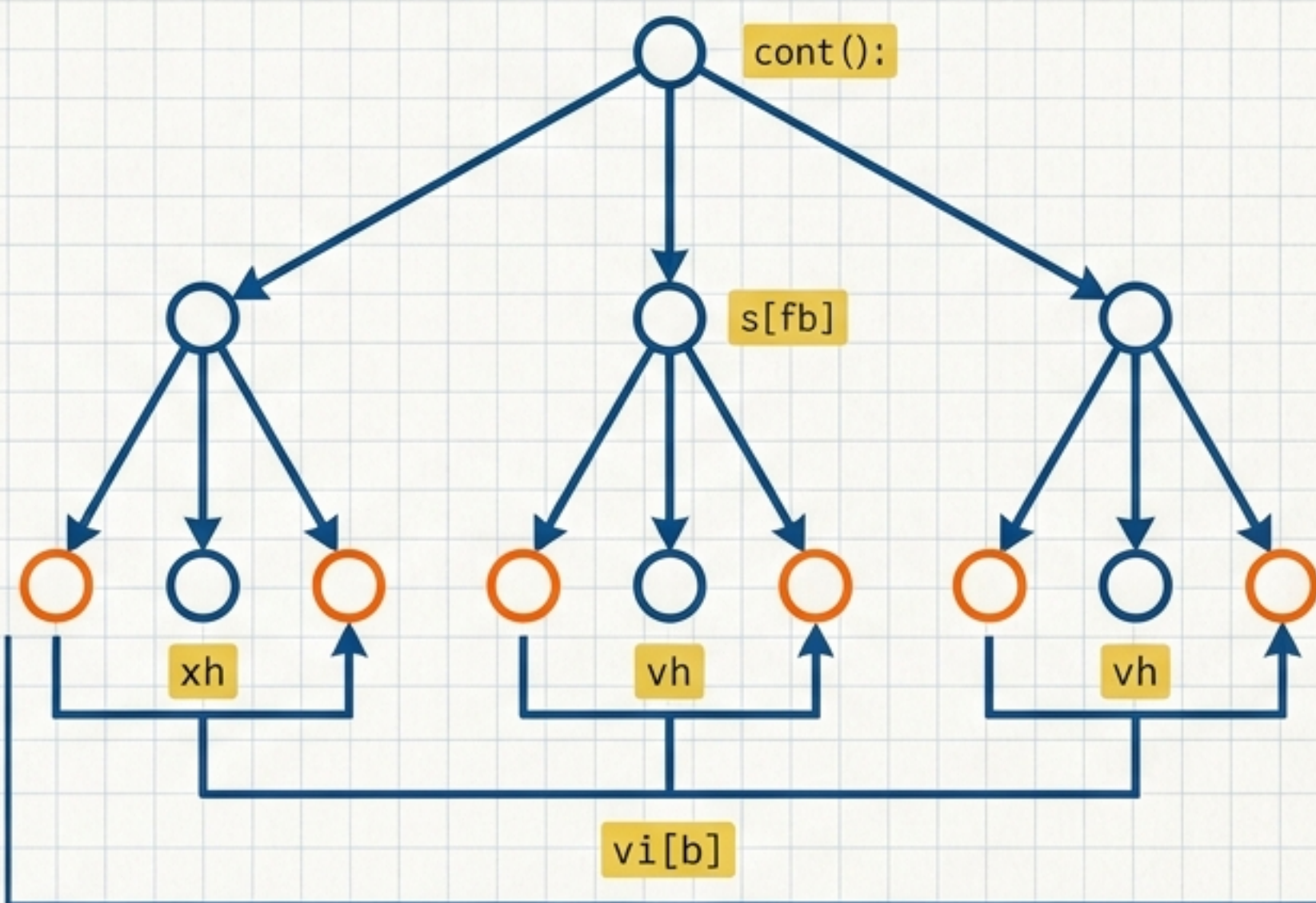
アルゴリズムの存在意義：計算量のブレイクスルー

巨大な桁数の乗算において、単純な筆算 ($O(N^2)$) は限界を迎える。カラツバ法は、掛け算を「ツリー状に分割」し、加減算に置き換えることで計算量を劇的に削減する手法である。本問はこの「ツリー構築」と「1次元配列への展開」が最大の焦点となる。

筆算の手法 ($O(N^2)$)



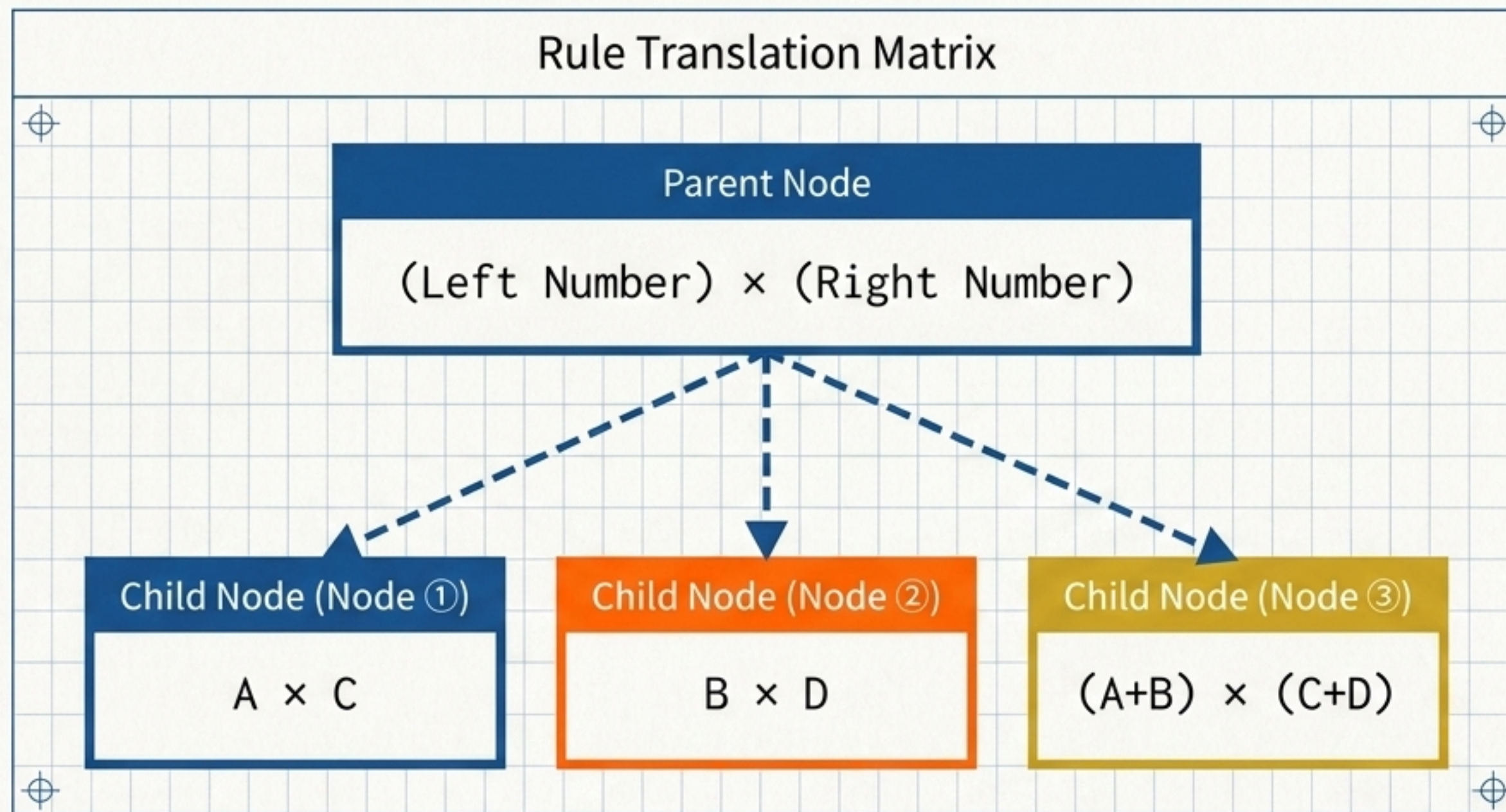
カラツバ法 ($O(N^{1.59})$)



ツリー分岐の絶対法則（基礎知識）

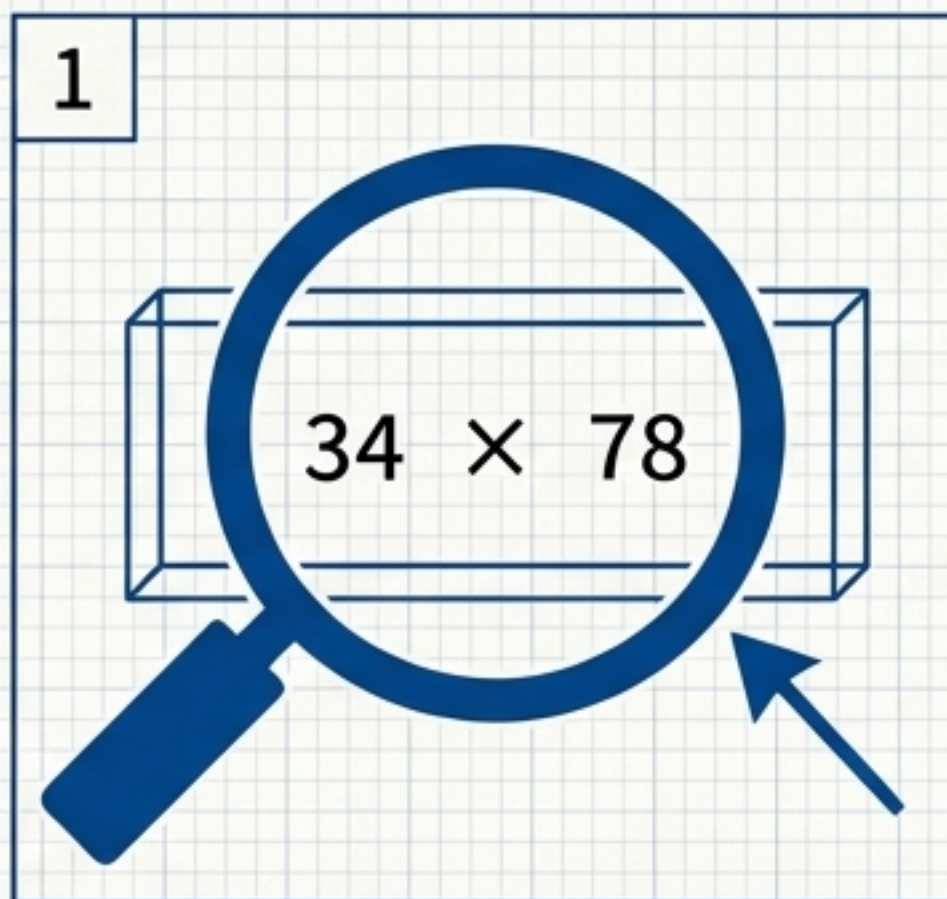
A, B, C, Dの定義を視覚化

A: 左の数値の上位桁
B: 左の数値の下位桁
C: 右の数値の上位桁
D: 右の数値の下位桁

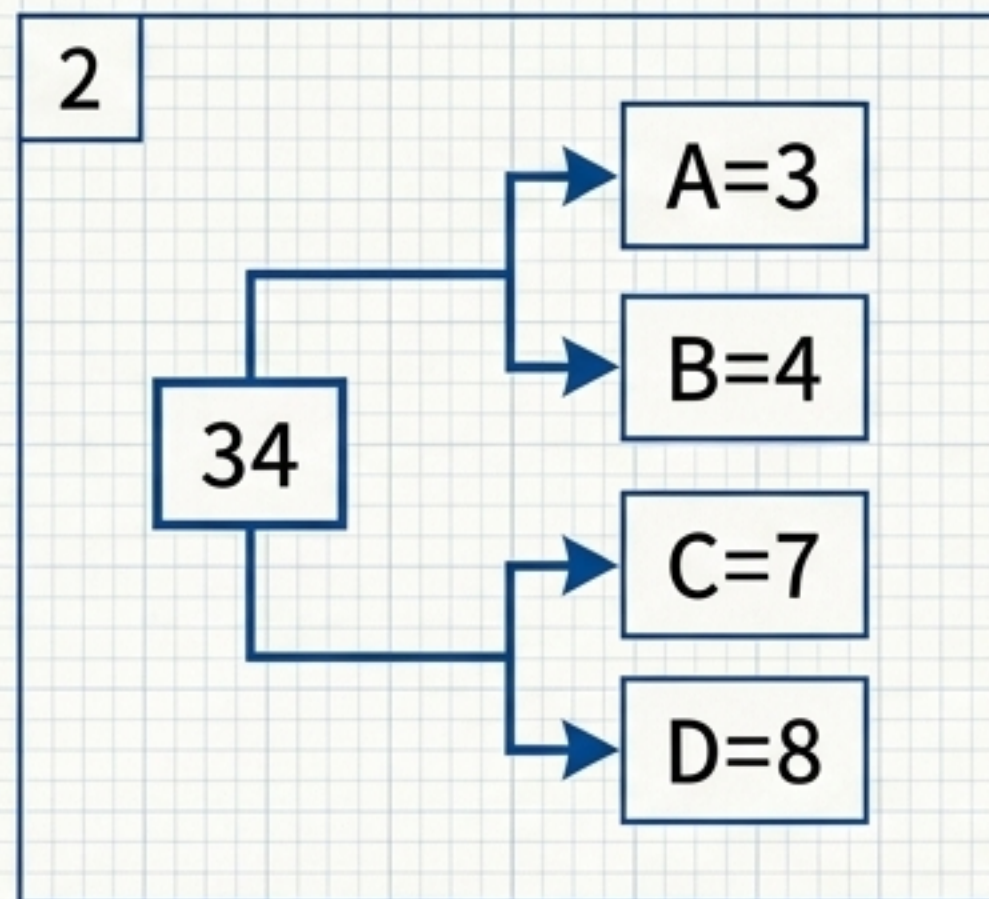


すべての設問は、この「1つの親が3つの子に分かれる」法則をプログラム上でどう管理しているか、という問いに帰着する。

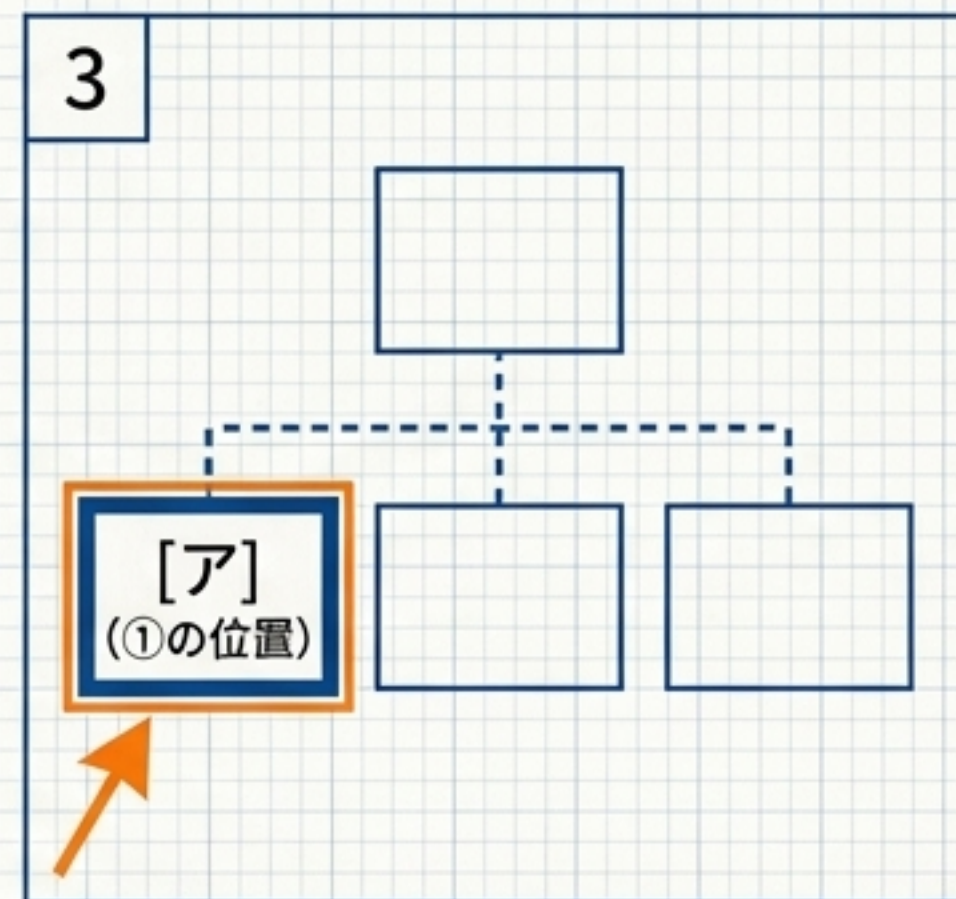
設問1 [ア] へのヒント：標準的なノード展開



Step 1: 対象ノード「34 × 78」を特定する。



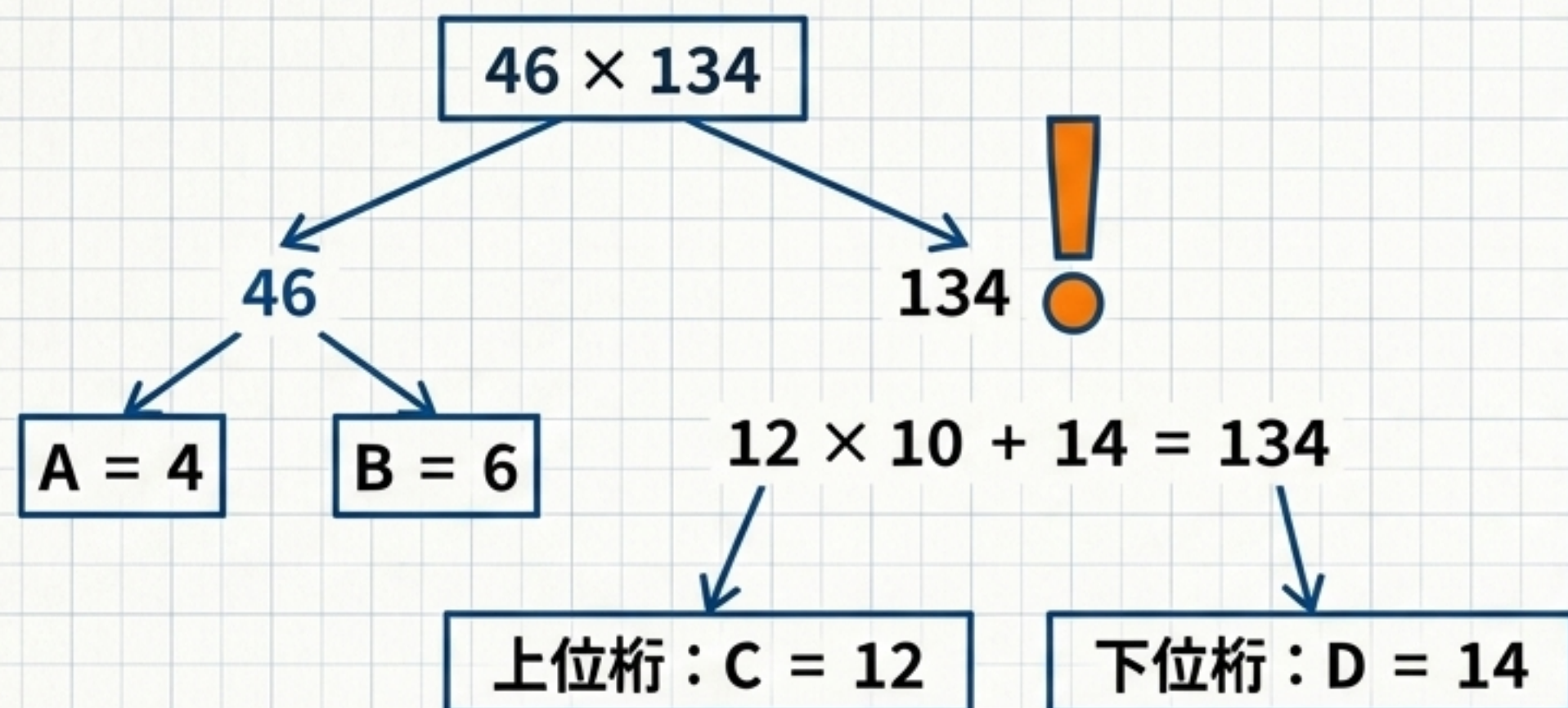
Step 2: 左の数値(34)と右の数値(78)を、ルールに従いA, B, C, Dに分解する。



Step 3: 空欄[ア]は「一番左 (①の位置)」である。

思考のプロセス：一番左の子ノード①の公式「 $A \times C$ 」に、分解した数値を当てはめるとどうなるか？

設問1 [イ] へのヒント：繰り上がりの「遅延トラップ」



[CAUTION]

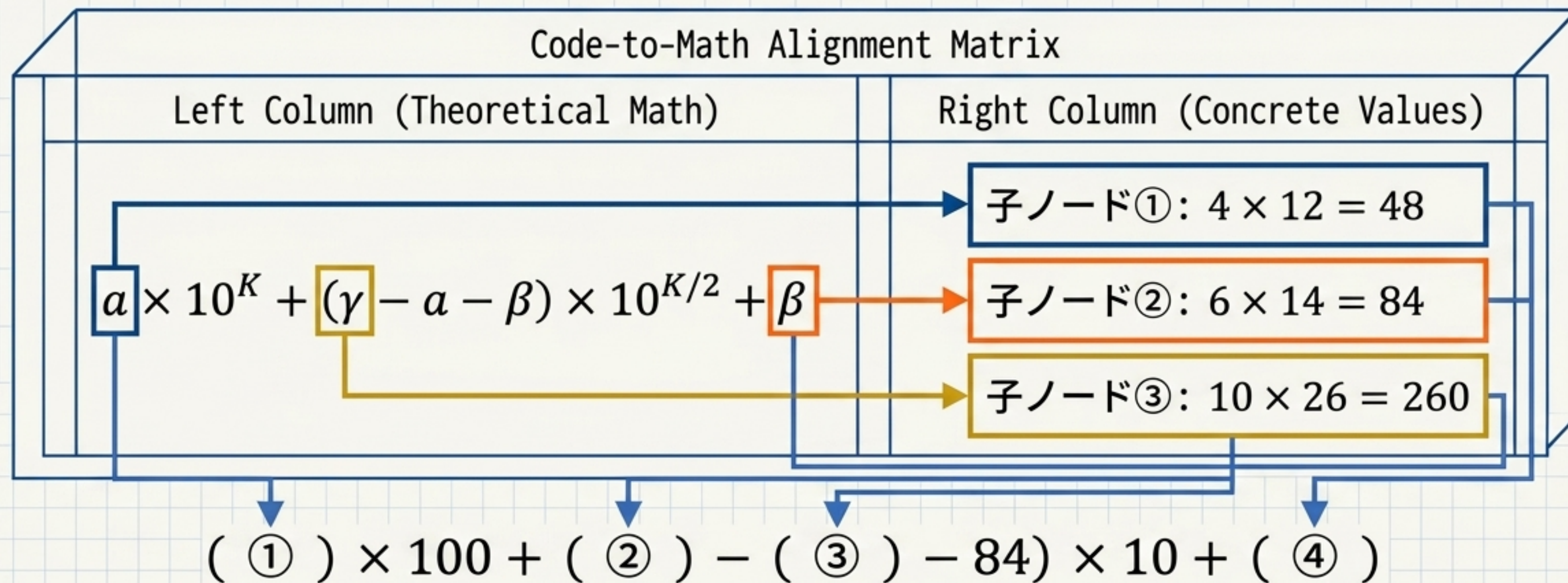
問題文の例外ルールに注意！
「繰り上がりの処理はツリーの最後でまとめて行う」

親から引き継いだ扱い（桁数M）は「2桁」である。そのため、「134」は3桁の数値ではなく、「10の位が12」「1の位が14」という異常な状態の2桁として処理される。

思考のプロセス：上位桁C=12、下位桁D=14とした場合、一番左のノード①（A×C）に入る式は何か？

設問2 へのヒント：数式とノードの完全マッピング

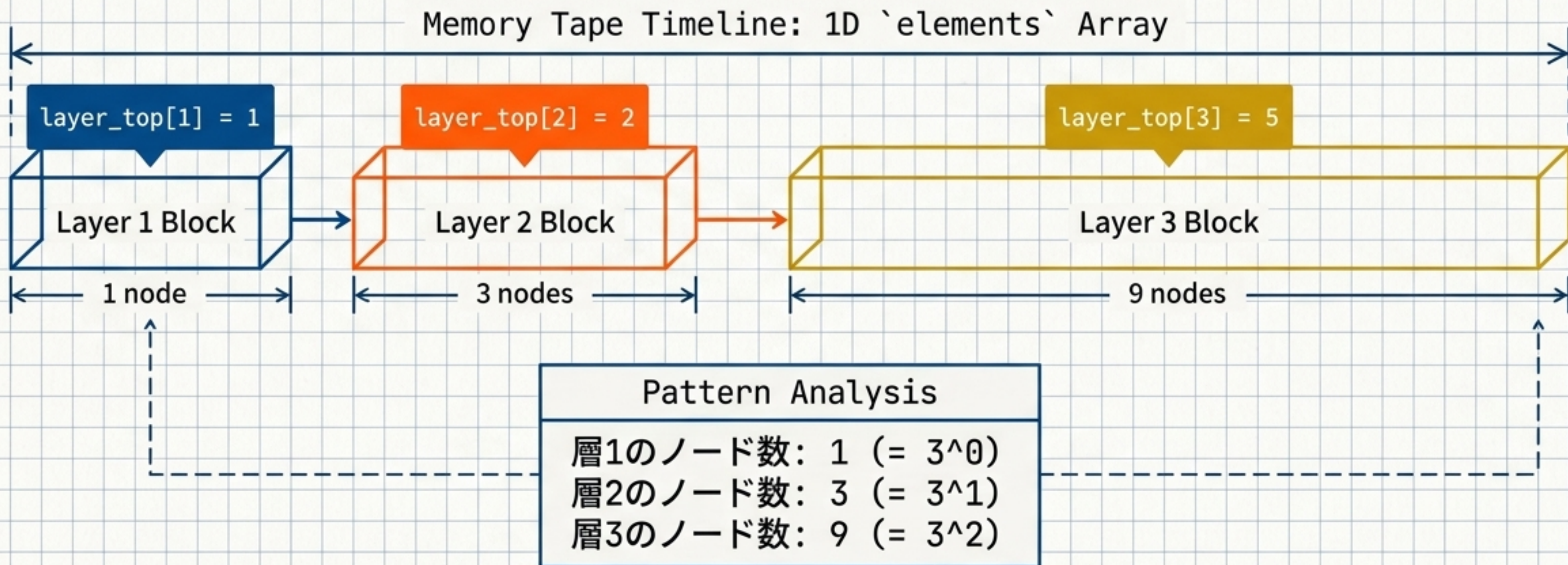
問題文に与えられた穴埋め用の式と、理論式を見比べる。



思考のプロセス：理論式の a, β, γ は、それぞれ子ノード①、②、③の計算結果に対応する。
色分けされた変数を、穴埋め式の対応する位置に代入せよ。

設問3 [ウ] へのヒント：2Dツリーの1D配列化（レイヤー管理）

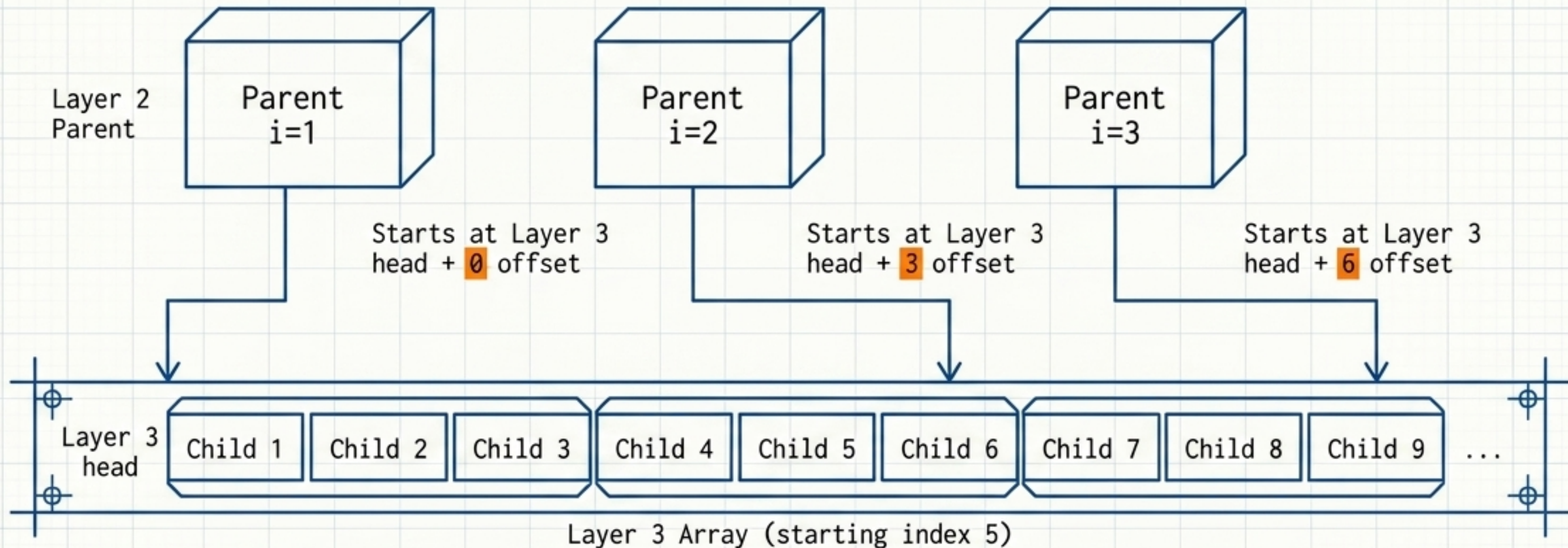
次の層の先頭インデックス
= 「現在の層の先頭インデックス」
+ 「現在の層のノード数」



思考のプロセス：各層のノード数は等比数列になる。表3のべき乗関数 $\text{pow}(m, k)$ を使用し、ループ変数 i を使って「現在の層のノード数」をどう表現するか？

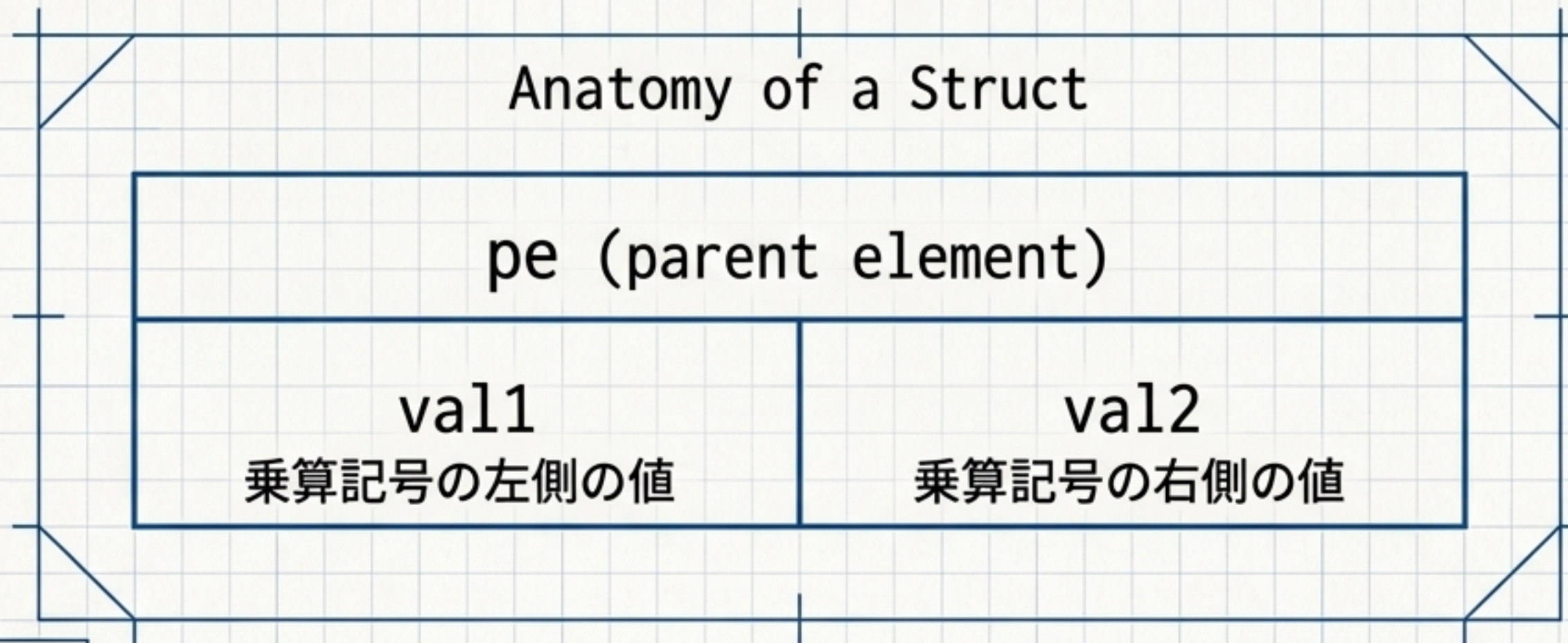
設問3 [エ] へのヒント：子ノードのオフセット計算

1つの親ノードから、常に「3つ」の子ノードが一気に生成され、配列に格納される。
そのため、親ノードが1つ進むごとに、子ノードの格納開始位置は右へズレていく。



思考のプロセス：「ズレ幅」は 0, 3, 6 ... と規則的に増加している。
この数列を、親ノードのループ変数 i を用いて数式化するとどうなるか？

設問4 [オ][カ] へのヒント：構造体からのデータ抽出



Logic Flow

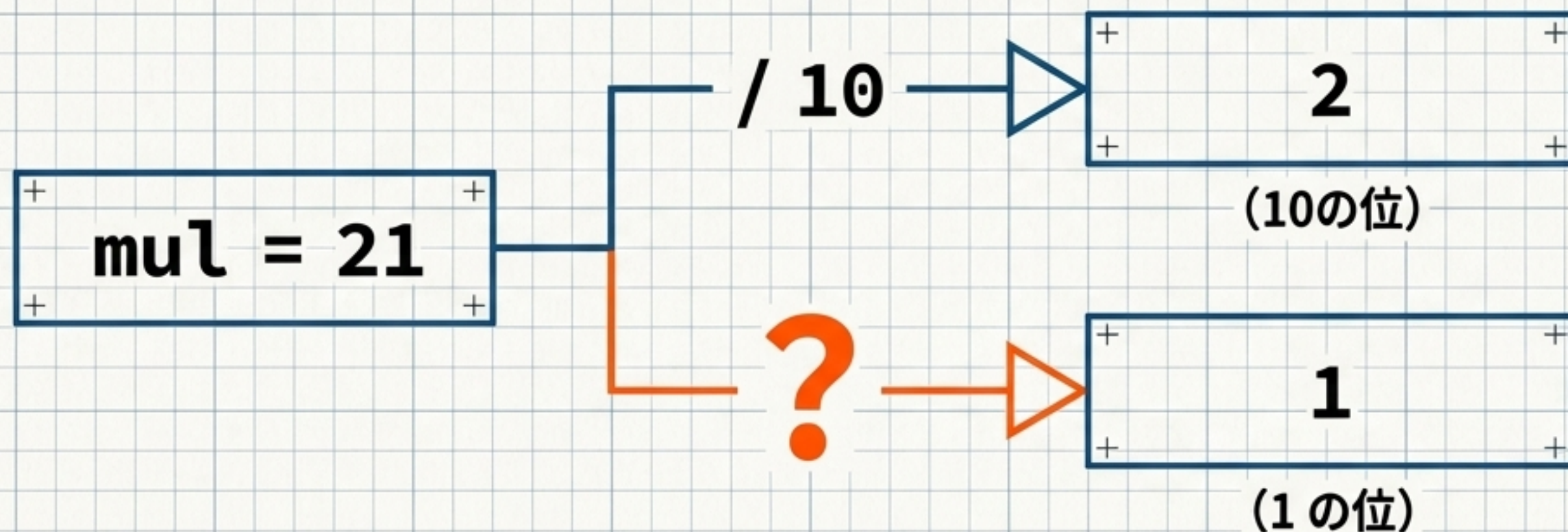
子ノード①の生成：親の左側の半分(A)と右側の半分(C)を抽出。プログラムでは、関数 left や right の引数として、親ノード pe の持つデータを渡している。

```
elements[tidx] ← new_elem(cn, left(?, cn), left(?, cn))
```

思考のプロセス：引数の左側（オ）には常に「親の左側の値」が、引数の右側（カ）には常に「親の右側の値」が入る。構造体のどの要素（プロパティ名）を指定すべきか？

設問4 [キ] へのヒント：最下層ノードでの桁分離

最下層の1桁同士の乗算結果（最大2桁）を、多倍長整数の配列の「1の位」と「10の位」に分離して格納する処理。コード上では、10の位を取得するために「`mul / 10`（整数除算）」がすでに記述されている。



思考のプロセス：1の位を取り出すには、10で割った「余り」を求める必要がある。
表3に用意されている関数を用いて、変数 `mul` に対する剰余演算をどう記述するか？

設問4 [ク][ケ] へのヒント： カラツバ法の核 (逆算ロジック)

ツリー構造上、生成された子ノードは
①(a)、②(β)、③(γ) の順に配列に並んでいる。

cidx = 子ノード①(a),
 $\text{cidx} + 1$ = 子ノード②(β),
 $\text{cidx} + 2$ = 子ノード③(γ)

[GOAL]
 s_2 の最終目的: $\gamma - a - \beta$

[LINE 2]
 $s_2 \leftarrow \text{sub}(s_1, \text{elements}[\text{cidx} + 1].\text{result})$ ← つまり $s_2 = s_1 - \beta$

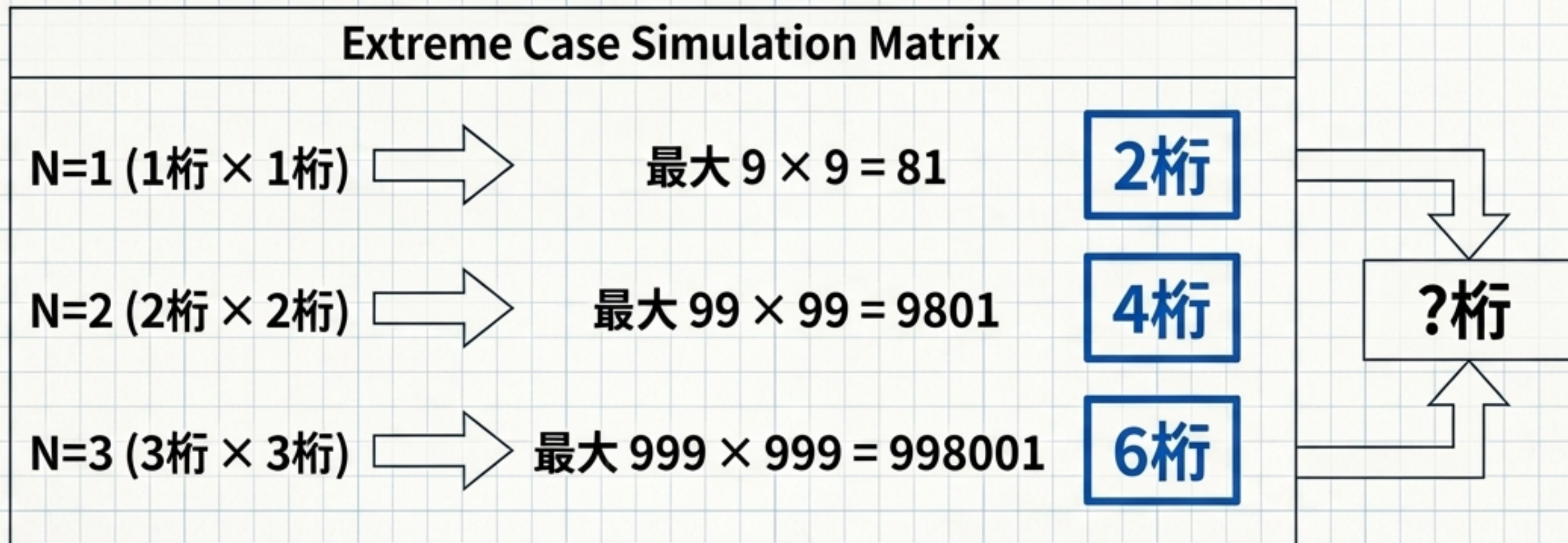
[DEDUCTION]
したがって、 s_1 は $(\gamma - a)$ でなければならない。

[LINE 1]
 $s_1 \leftarrow \text{sub}([\text{ク}], [\text{ケ}])$ つまり $\gamma - a$ を実現する

思考のプロセス：関数 $\text{sub}(p, q)$ は $p - q$ を行う。目的の $s_1 = \gamma - a$ を実現するためには、引数 [ク] と [ケ] に、それぞれのインデックスの要素を指定すべきか？

設問5 へのヒント：配列要素数の最大境界（ワーストケース）

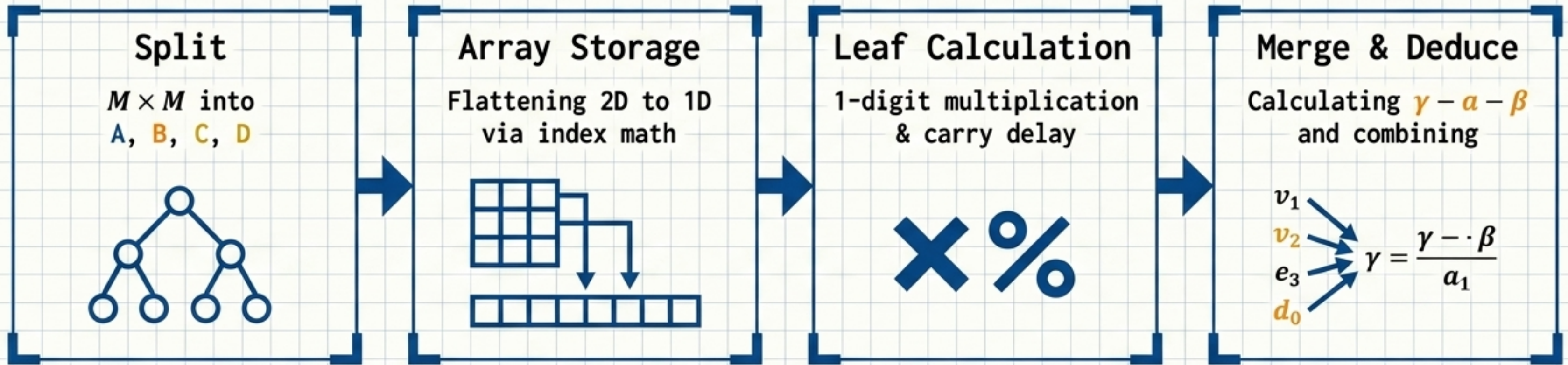
構造体の要素 values は、計算結果の桁ごとの値を管理する 1 次元配列である。
「配列の要素数 = 計算結果の桁数」となる。



思考のプロセス：数学の一般法則として、N桁の整数同士を掛け合わせたとき、桁数が最大に膨れ上がる「各桁が9」の極端なケースを想定する。結果は最大で何桁になるか？Nを用いた数式で表せ。

SYNTHESIS：アルゴリズムの全容

The Karatsuba Blueprint



孤立して見えた各設問の空欄は、巨大なデータを分割し（Divide）、計算し、再び統合する（Conquer）という一つの美しく最適化された「設計図」の構成要素である。

最終チェックリスト：解答導出へのキーポイント

- ☐ ツリーの分岐ルールと A, B, C, D の対応関係を視覚化できているか
- ☐ 繰り上がり処理の「遅延」ルールによる特殊な数値表現を理解したか
- ☐ 等比数列とループ変数 i を用いたインデックスの数式化ができるか
- ☐ sub 関数の逆算から $\gamma - a - \beta$ の構造を読み解けたか
- ☐ ワーストケース（最大値）を想定した物理メモリの境界を把握したか

「答えはすべて問題文のルールの中に隠されている。
ヒントを元に、アルゴリズムの設計図を自らの手でトレースしよう。」