コーディング課題

プログラムの実行方法

それぞれのプログラムの実行はシェル上で(x は数字)

./qx.py　　検証用モードのときには　　./qx.py validataion

入力ファイル qx_in.txt は同じディレクトリにあるものとし、qx_out.txt が同じ場所に出力される。検証用の動作は以下のそれぞれのプログラムの説明に記載されている。

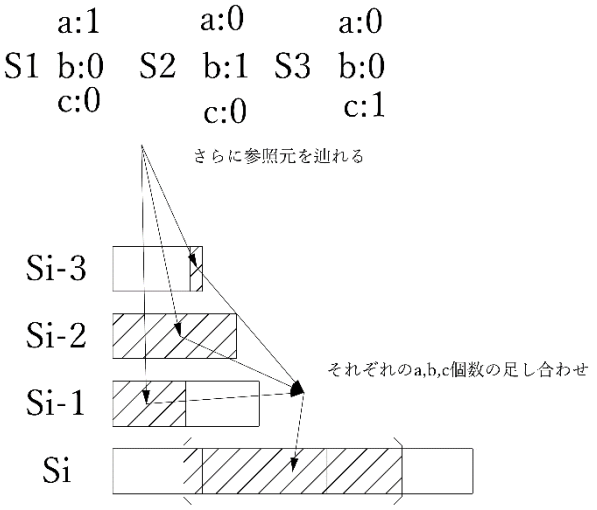
問 1

x と y が与えられたとして 3x3 行列に含まれる 9 個の各要素($i_1, \cdots, i_9 \in \{x, y\}$)の全組み合わせ2^9通りに対して総当りで行列式を計算し、目的の行列式の値となる行列を数え上げる。行列式の計算には Numpy の関数を用いている。この場合、結果は浮動小数点で得られるため整数へと丸めているが、たかだか $|x|, |y| \leq 10$の 3 乗程度の計算であるので誤差は 1 にくらべてはるかに小さく、計算値には影響はないと考えられる。事実、$|x|, |y| \leq 10$の条件を満たす全ての整数入力の中で丸め誤差の最大値は 3.183231456205249e-12 であった。→検証用モードで再現可能

問 2

S_iは直前 3 つの文字列をつなげたものなので、ある区間での a,b,c 数はつなげる前のそれぞれの$S_{i-1}, S_{i-2}, S_{i-3}$で対応する区間で求めた a,b,c 数の総和と等しい。前のS_iへの参照を繰り返していくことで、最終的にはS_1, S_2, S_3にたどり着く。これを利用して再帰的に次のようなアルゴリズムを用いて計算することができる。以下はいま考えているのがS_iであるとして

1. $i \leq 3$であれば、(a,b,c) = (1,0,0),(0,1,0),(0,0,1)のどれかを通知して、参照元に伝える。
2. そうでなければ$S_{i-1}, S_{i-2}, S_{i-3}$それぞれの対応区間に分解し、それぞれの値の集計(→1.)を任せて各値が求まったら集計して、参照元に伝える。

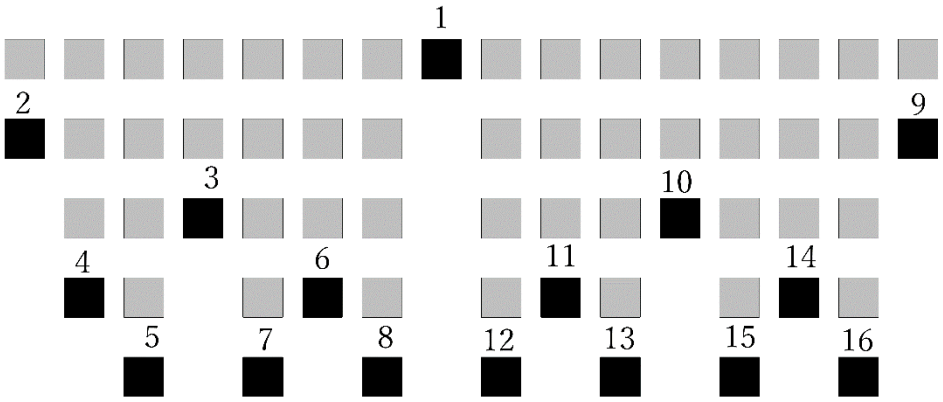


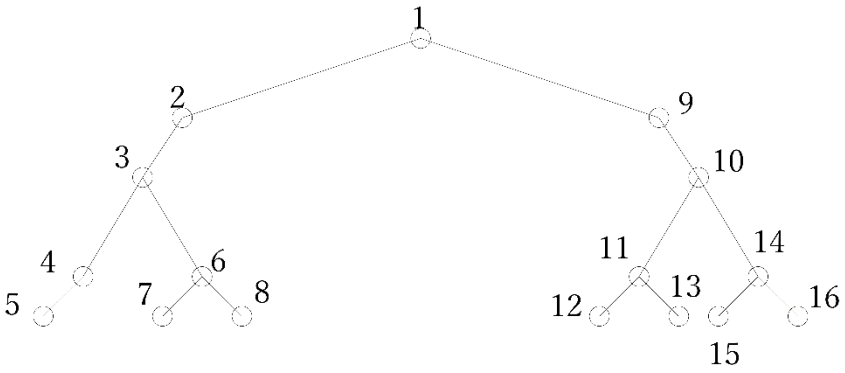
しかしながら、ただこれを実装しただけでは遅いのですでに計算した各S_iに含まれている a,b,c 数をメモ化する。このプログラムの動作検証は他のプログラムの出力結果と比較することで行った。具体的には文字列の連結を繰り返すことで、実際にS_iを生成し愚直に数えるプログラムを作成した。自分が確認できた範囲ではS_{15}までの文字列で、取りうる全ての p-q 区間で検証した結果、出力結果は全て一致した。

→検証用モードで再現可能

再帰関数的なプロセスであるので i=15 というある程度十分な分だけで正しい解が生成できていることから、i を大きくしていっても正しい解が得られていると考えられる。

問 3





専有される椅子は右の図で上から下の順番で決定される。空間のなかで椅子を専有することで左右の空間を区切り、また区切られた区間で同様のことを繰り返していくことで、逐次的に決定できる。したがって木構造的な関係性を作ることができる(親が専有されなければ、子は専有されない)。それに対して直系の親子関係になければ階層の順序は前後してもよく、そのときはそのノードのスコアとして区間の幅から計算できる他の専有椅子からの距離が最も大きくなる椅子を専有する。これを繰り返していくことで、最終的に全ての椅子が順番に専有される。

アルゴリズムについて

1. 初めに与えられた専有椅子の位置から左右の区間およびその区間で最初に専有される位置を決めることを繰り返してき、木を作っていく。専有される椅子を決めると同時にその区間が設定された状態での他の椅子との距離をスコアとして決定する。
2. つぎにスコアを図の番号順のように左側から深さ優先で配列に並べていく。
3. それをソートすることによって、最大のスコアを優先しながら親子の順番を破らないように専有される椅子の順番が決まる。

検証について

愚直に全てのいちで他の椅子からの距離を示す値を場のように保持しておき、新たな椅子が専有されると周りの値を更新していくこと、また専有はそのときに最も高い値を持つ位置で起きるというルールでプログラムを作成した。その結果と比較することで検証した。その結果 n=100 までは全ての a1 で一致することを確認した。→検証モード