スーパーコン 2010 本選問題：戦略のヒント

今年の本選問題は数を計算する問題です．数を計算する問題の解法として代表的なものに「探索木による列挙」と「動的計画法」があります．以下では，その 2 つを説明していきます.

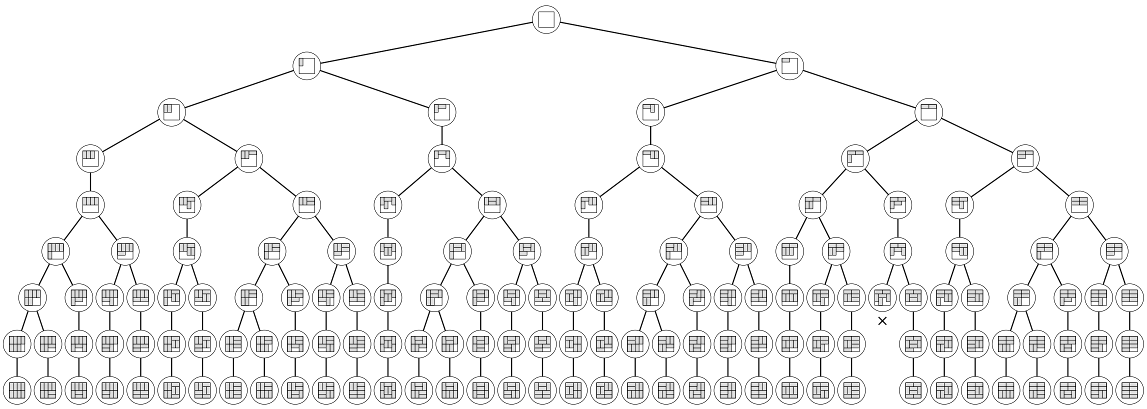
説明のために，本選問題課題の中にある「レンガ」,「広場」,「障害物」ということばをそのまま用いていきます.

注意： ここで説明されることを鵜呑みにしてはいけないかもしれません．重要なことは自分たちの頭と手で創意工夫をすることです．プログラムやアルゴリズムに「こうしなくてはならない」という決まりは文法規則以外にありません．あるのは慣習や原則だけです．慣習・原則にとらわれたくない場合は，ここから始まる説明を読まず，この紙をすぐに破り捨てましょう．ただ，慣習・原則は先人が築いてきた知恵の結晶でもあります．その中には役に立つことが含まれている可能性もある，ということは心に留めておいて下さい.

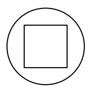
1　探索木による列挙

数を計算する方法として**列挙**があります．これはすなわち，敷き詰め方の 1 つ 1 つを全部見つけていくという方法です．そのための汎用的方法の 1 つが**探索木による列挙**です.

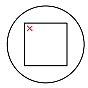
まず，探索木の例を示します.



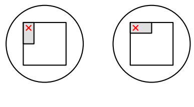
この図の見方を説明します．一番上に 4×4 の正方形が円に囲まれて描いてあります．これは $m = n = 4$ で，障害物のない問題が与えられている状況を表しています.



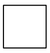
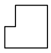
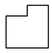
この広場に対する敷き詰め方の総数を計算するために，左上隅 (次の図で×の印が付けられた場所) を覆うレンガがどのように置かれるか考えます.



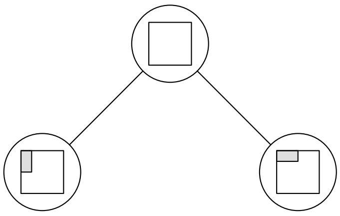
それは，次の図のように，縦か横にしか置けません.



このことから,

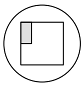
 の敷き詰め方の総数 =  の敷き詰め方の総数 +  の敷き詰め方の総数

となることが分かります．これを最初の図では

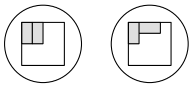


という部分で表現しています.

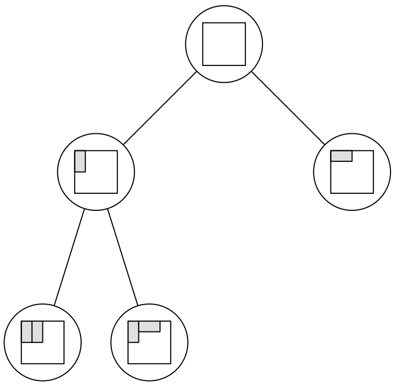
では，この左側にある



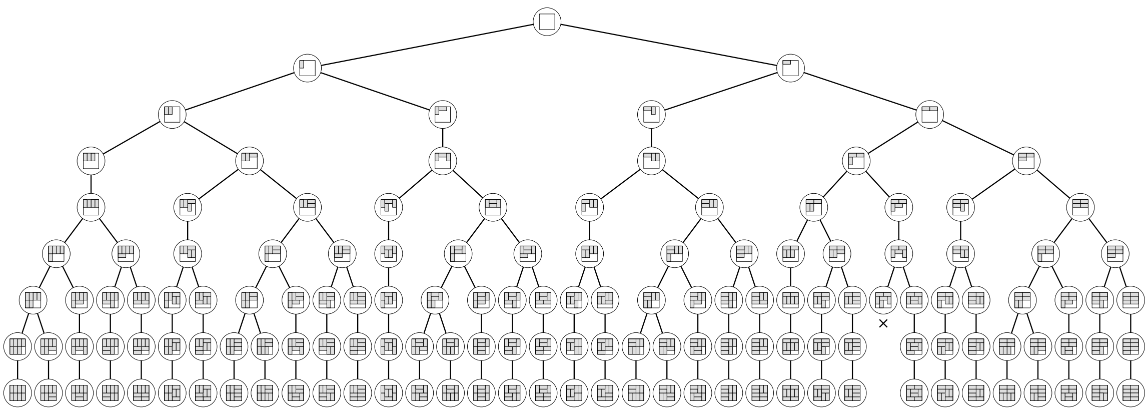
がどのように敷き詰められるかを同様に考えます．左上の隅を×だと思うと，次のように分けられます.



したがって，先ほどの表現を用いると次の図のように状況が表現できます.

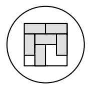


この操作をでき上がるすべての状況に適用したものが始めに挙げた探索木です．もう一度示します.



これが木と呼ばれている理由は，公園に植えてある木の上下を逆にしたものに似ているからです．丸で囲まれたもの１つ１つを**ノード**と呼んでいます．それよりも下の部分がないノードは**葉**と呼ばれます．

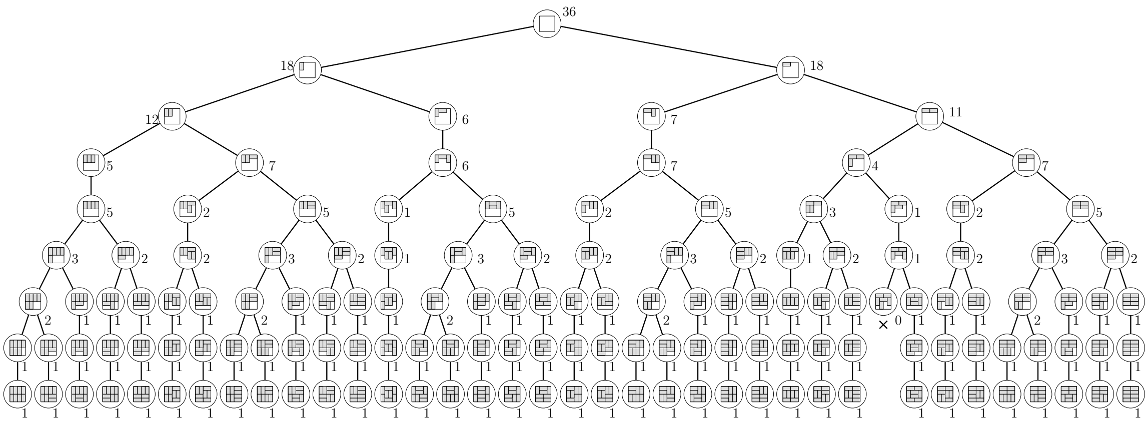
葉には２種類あります．１つは広場全体がレンガで埋め尽くされた状況を表しているものです．もう１つは，広場全体がレンガで埋め尽くされているわけではないですが，最終的にレンガで埋め尽くすことができない状況を表しているものです．上の探索木では後者の葉が１つだけあります．次のようなものです．



この状況では，左下にある空白１つをどのようにしても埋めることができないわけです．探索木の図の中で，この事実は「×」として葉の下に示してあります．

さて，その２種類の葉に対して，その状況を敷き詰める方法の総数はいくつでしょうか．それについて，「広場全体が埋め尽くされている状況」を敷き詰める方法は「1」であり,「広場全体が埋め尽くせない状況」を敷き詰める方法は「0」です．これで，葉については敷き詰め方の総数が分かりました．

では，葉ではないノードが表す状況に対する敷き詰め方の総数はいくつでしょうか．それは，そのノードの直下にあるノード (よく「子ノード」と呼ばれます) に対する敷き詰め方の総数の和 (足し算の結果) となります．つまり，葉に近い方から順に敷き詰め方の総数を足していけば，最終的に広場全体の敷き詰め方の総数が分かるわけです．探索木の各ノードが表す状況に対する敷き詰め方の総数を，そのノードのそばに書いたものが次の図です．

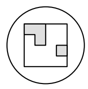


図の一番上にある，$m = n = 4$ で障害物がない状況に対して，それを敷き詰める方法の総数が正しく 36 になることが確認できます．

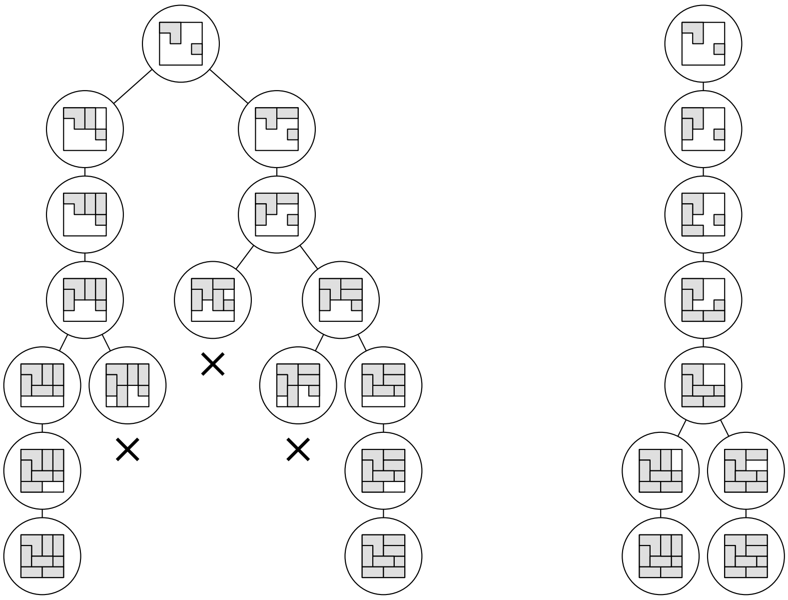
サンプル・プログラムは，この考え方に沿って再帰的に構成しています．

この考え方に沿ったアルゴリズムでは，探索木のノードの数が計算時間に大きく反映され，ノードが少なければ少ないほど計算時間が短くなると考えてよいです．ノードの数を小さくする方法の１つに，レンガを置く場所の工夫があります．

例えば，次のような広場を考えましょう．



「探索木による列挙」によってこの広場を敷き詰める方法の総数を数えるとき，レンガを置く場所によって次の図の左と右にあるように探索木のノードの数が大きく変わってきます．

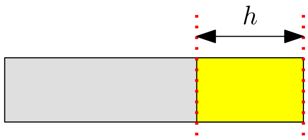


この場合，右の探索木の方がノードが少なく,「×」の付く葉もなく，敷き詰め方を無駄なく見つけていることが分かります．この例は $m = n = 4$ と小さいため，威力が分かりにくいですが，m や n が大きな問題に対してはどこから埋めていくのか考えることはとても重要になってきそうです．

2 動的計画法

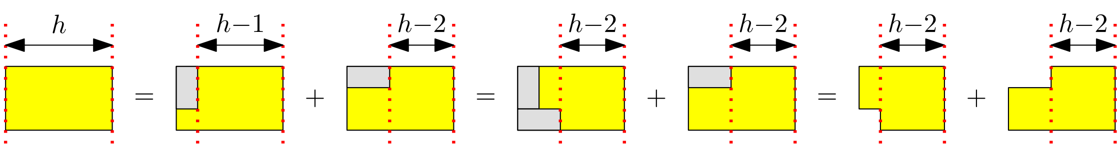
予選問題では，$2 \times n$，$3 \times n$ の障害物のない広場にレンガを敷き詰めることを考えました．そのために，多くのチームが漸化式 (再帰式) を導いていました．これの基礎となる考え方は m が 2 や 3 でなくても適用可能であり，また，障害物が存在していても可能です．実際に $m = 3$ の場合で，障害物がない場合の考え方を復習しましょう．

右から h 個の列までを敷き詰める方法の総数を考えましょう．つまり，次の図の黄色い部分の敷き詰め方の総数です．



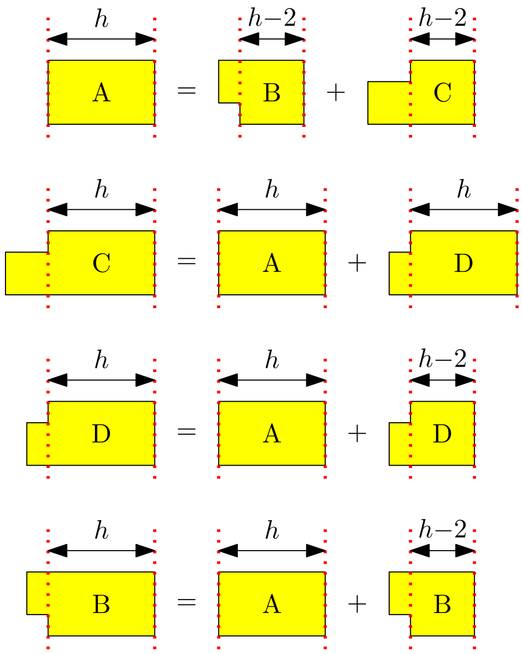
これで，$h = n$ のとき，広場全体をレンガで敷き詰める方法を数え上げることができます．

先ほどの「探索木による列挙」の考え方を使うと，この部分の敷き詰め方の総数は次のように計算できます．



つまり，右から $h-2$ 個の列までを完全に含み，そこから列 2 つ分だけ左へ規則的にはみだしている広場に対する敷き詰め方の総数が分かれば，計算できるというわけです．

この式の右辺にある広場に対しても同じ考え方で計算をすると次のような式が得られます．



アルファベットの A から D は，左へのはみだし方が同じ広場を同じ文字で表すように付けられています．このようなパターンを h が小さい方から順に右から計算していけば，最終的に $h = n$ となるような A パターンの敷き詰め方の総数が計算すべき出力となります．

この考え方は縦の長さが 3 でなくても有効であり，また，障害物があっても有効です．そのような場合に対して，実際にどのようなアルゴリズム，プログラムを作ればよいのか考えることが重要な点になります．

注意点が 1 つあります．予選問題の場合，つまり，障害物がない場合では，上のような A から D で示されたパターンと右からの列数 h によって，きれいな漸化式 (再帰式) を導くことができました．しかし，障害物があると，そのような漸化式を導くことは不可能かもしれません．漸化式を導くことに主眼を置かず，どのような原理で計算をすれば，出力すべきものが得られるのかということを考えてみて下さい．

3 その他の情報

3.1 どちらが「お得」か？

「探索木による列挙」と「動的計画法」が基本的な戦略になるのですが，この 2 つは一長一短です．問題作成委員会がいろいろ試してみたところ，次のような特徴がありそうだと分かりました．

- 「探索木による列挙」は障害物が多ければ，m や n が大きな問題でも高速に解ける．しかし，障害物が少ないと，m や n が小さな問題でも解くことが難しくなる．
- 「動的計画法」は障害物の数に依存せず，m が小さな問題ならば高速に解ける．しかし，m が大きいと，解くことが難しくなる．

そのため，この 2 つの基本的戦略をどのように組み合わせていくのかが鍵になるかもしれません．

3.2 剰余演算に関する注意

0 以上の整数 a と正整数 k に対して，$a \% k$ と書いたら「a を k で割った余り」を表すこととします．このとき，次が成り立ちます．

$$
\begin{aligned}
(a+b) \% k &= ((a \% k) + (b \% k)) \% k, \\
(a \times b) \% k &= ((a \% k) \times (b \% k)) \% k.
\end{aligned}
$$

ただし，a と b は 0 以上の整数，k は正整数であるとします．

3.3 最後に

問題数は全部で 100000 (十万) で，問題の種類も多様です．最終トライアルではすべての問題を解こうとはせずに，解ける問題を見つけて解いていくということが重要かもしれません．そのために「`statistics.txt`」の情報をどのように使えそうなのか，ということを考えることも重要かもしれません．

3.4 付録

広場が正方形であり (つまり，$m = n$ であり)，障害物が全く存在しない場合の敷き詰め方の数を次のページに挙げます．これは正しいものであるとして，プログラムを作成してもよいです．ただし，この情報が役に立つかどうかは分かりません．参考にして下さい．(より大きな n の値に対する敷き詰め方の数は「内部向けノート」のページに掲載します．)

```
n 敷き詰め方の総数
2 2
4 36
6 6728
8 12988816
10 258584046368
12 53060477521960000
14 112202208776036178000000
16 2444888770250892795802079170816
18 548943583215388338077567813208427340288
20 126998401125623583424260275310229393429857624985 6
22 3027370049223160301515020410877256278093158742819351453900 8
24 743577567912292296514193039416768412010553529732230861356917352710553 6
26 188182184808134567624218569639992866504751558980669552282959258067778840281537003 52
28 490705382430044396584844668594853686545442654322252039155636063994159671359376052625754522402816
30 1318415454722440274064961887579123753638916964432212796946269479121884599564377001055717733349003602949120000 00
```