


注意！この記事を書いているのは水色です！

CAUTION This article is written by MIZUIRO

くそでか注意

- 飽きました
- 途中までしか記事書いてません
- 参考にならない可能性があります。

 コンテスト成績証

ユーザ名 **nibosea**





コンテスト名 AtCoder Beginner Contest 252

順位 **552nd** / 9995

パフォーマンス **1766**

レーティング **1196** → **1268** (+72)

発行日: 2022/05/21



[niboseaさんのコンテスト成績表]

大勝利して,水色に戻りました。おめでとうございます。ありがとうございます。

G問題、 $O(N^3)$ が通りそうだな、とは思ったがこんなの解けねえよバーカ！！

長い時間熟成してようやく理解...というか、たしかにそうやれば解けるなっていうのが分かったのでシェアしたいと思います。

- 自己満記事です
- 理解できなくても文句言わないでください
- この記事で人々の理解の一助になれば嬉しい

[CODE FESTIVAL 2015 決勝 G-スタンプラリー](#)は類題デス。っていうか制約違うバージョンデス。っていうかこれ,Pre-orderのACコード投げたら多分通ります。。。草

リセット 検索									
提出日時	問題	ユーザ	言語	得点	コード長	結果	実行時間	メモリ	
2022-05-23 18:17:56	G-スタンプラリー	nibosea 	C++(GCC 9.2.1)	0	2031 Byte	WA	36 ms	10324 KB	詳細

通りませんでした...

問題文を見てみるとmod1e9+7でした。そりゃ通らないっすね。

提出日時	問題	ユーザ	言語	得点	コード長	結果	実行時間	メモリ	
2022-05-23 18:38:57	G- スタンブラリー	nibosea	C++ (GCC 9.2.1)	100	1987 Byte	AC	32 ms	10300 KB	詳細
2022-05-23 18:25:05	G- スタンブラリー	nibosea	C++ (GCC 9.2.1)	0	1942 Byte	WA	39 ms	10360 KB	詳細
2022-05-23 18:20:28	G- スタンブラリー	nibosea	C++ (GCC 9.2.1)	0	2032 Byte	WA	33 ms	10360 KB	詳細
2022-05-23 18:19:35	G- スタンブラリー	nibosea	C++ (GCC 9.2.1)	0	2031 Byte		CE		詳細
2022-05-23 18:17:56	G- スタンブラリー	nibosea	C++ (GCC 9.2.1)	0	2031 Byte	WA	36 ms	10324 KB	詳細

1

modを直しても通らなかった、c[0]が1じゃないケースがあって死んでた。

```
c[0] != 1 cout << 0 << endl;
```

をしたら通った

いや、本題はこれじゃねえんだよ

[ABC252 G- Pre-Order](#)

[区間DP,行きがけ順,黄diff]

1. [merom686さんのブログ](#)を読んで見る。なんか分かりそうな気がする、雑にコードを書いてみる
がサンプルが合わない。
2. [公式解説](#)を読む。わからん。わからない。
3. [公式解説動画](#)を見る。わからん。わからない。
 - ここでMr.Snukeが、CODEFESTIVALに類題があったとかいうので探す
4. [類題の解説](#)を読む
 - これのp36~
 - うーん？なるほど？
 - とりあえずDPテーブルを手でかいてみる。
 - なんとなく分かった
 - コード書いてみる
 - AC
 - あーにや、解説記事書きます
- 前提知識
 - 木とか森とかなんとなく理解、DPっぽいなってこともなんとなく理解
 - 問題の意味も理解している
 - まあ、要するになんとなくやりたいことは分かっているが理解するのだるいなって人向け

とりあえず、先程のスライドのp44の写真を取ってきたいのですが、写真はつけて許してくれるか
分からないので、手で写経します。。。と言おうとしたのですが、Latexの数式の書き方がわからな
かったので写真張ります。許してクレメンス。

解法

- 区間DPを計算する
- 状態
 - $DP_tree[L][R]$
 - $C_L \sim C_R$ からなる木 (C_L が根となる) が何通りあるか
 - $DP_forest[L][R]$
 - $C_L \sim C_R$ からなる森 (C_L が最も左の木の根となる) が何通りあるか
- 遷移
 - $DP_tree[L][R] = DP_forest[L+1][R]$
 - 森に根をくっつける
 - $DP_forest[L][R] =$ 以下のsum
 - $DP_tree[L][i-1] * DP_forest[i][R]$
 - $L < i \leq R$ かつ $C_L < C_i$ を満たすような i について
 - 森の左に木をくっつける
 - $DP_tree[L][R]$
 - 木は森である

< 44 of 71 >



(参照 <https://www.slideshare.net/chokudai/code-festival-2015-final>, 2015-11-15, AtCoder Inc)

ということで、この解法の遷移式をもとに、手を動かして理解を深めることにしていこう。

サンプルは

```
N = 5
C = {1, 3, 5, 4, 2}
```

問題だとAだけど、上のスライドにあわせるためにCにするね

1. 遷移1つ目 $DP_{tree}[L][R] = DP_{forest}[L+1][R]$
2. 遷移2つ目 $DP_{forest}[L][R] =$ 以下のsum
 1. $DP_{tree}[L][i-1] * DP_{forest}[i][R] (L < i \leq R \text{ かつ } C_L < C_i)$
 2. $DP_{tree}[L][R]$

って感じらしいデス。

なるほどね

あ、初期化が $DP_{tree}[i][i] = 1$ らしいです。

(1の遷移は、ただ単に森のDP配列の値が、木のDP配列の1個上の場所に移動する...ってだけっすね)

とりあえず初期化します。

!(/home/kinjo/.config/Typora/typora-user-images/image-20220523200655822.png)

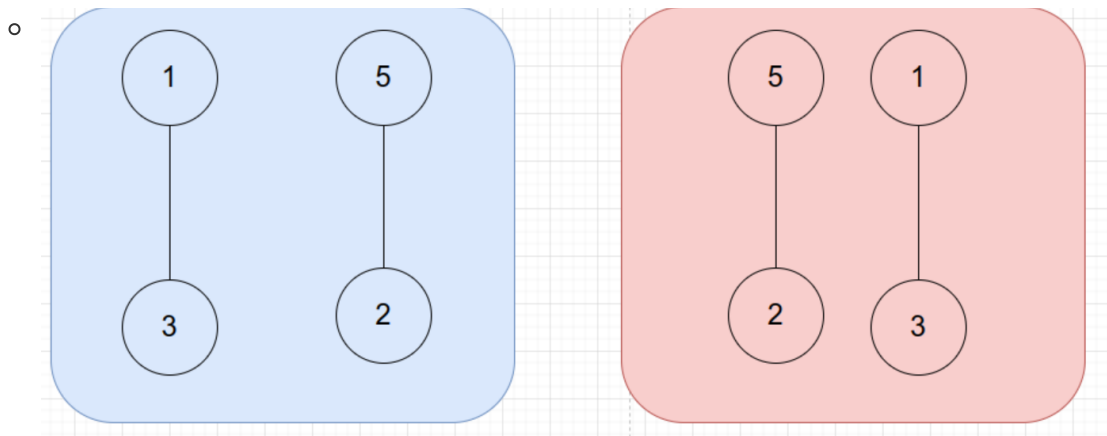
DP Tree					
	0	1	2	3	4
0					
1					
2					
3					
4					

DP Forest					
	0	1	2	3	4
0					
1					
2					
3					
4					

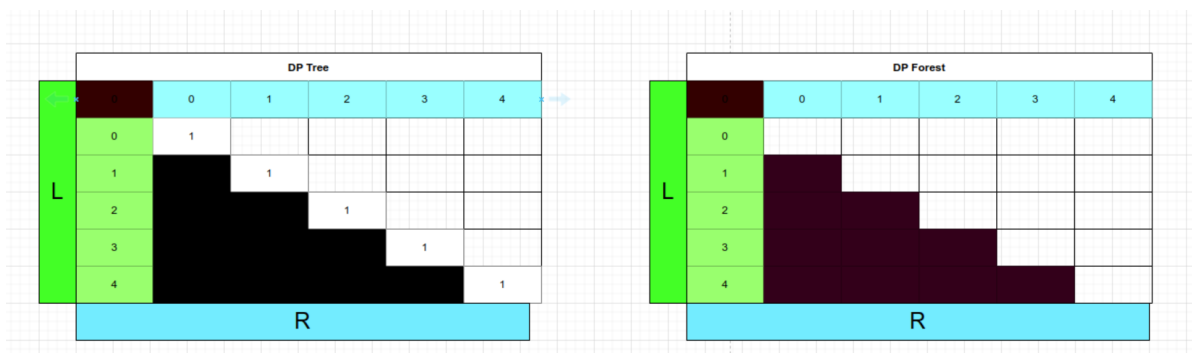
ホイ。

忘れてたけど、多少解説します。

- $DP_{tree}[L][R] := C[L]$ を根としたときに、 $C[L], C[L+1], \dots, C[R-1], C[R]$ からなる木の場合の数
 - 確かに $DP_{tree}[0][0]$ は、 $C[0] = 1$ のみを頂点に持つ木なので、1通りしか無い
 - $DP_{tree}[1][1]$ は、 $C[1] = 3$ のみを頂点に持つ木なので、1通りしか無い
 - 他の位置についても同じ。
- っていうのを踏まえると、初期化の意味も納得出来るだろう。
- ついでに forestのDP配列も定義をするね
- $DP_{forest}[L][R] := C[L], C[L+1], \dots, C[R-1], C[R]$ からなる森の場合の数で、 $C[L]$ が最も左の木の根となるものの場合の数
 - もっともひだりの木の根...??って感じかもしれないので、書くね



- 青い方はOKだけど、赤い方はNGです
- どうしてこういうふうにするかっていうと、よく分からないけど、こうすることで重複が無くなったり、正しく条件を満たすものが数え上げられるからだと思う。
 - なんか、赤い方の森を作ってしまったときに、頂点0(本当はないけど)の子供として[5-2]と[1-2]をくっつけたときに、行きがけ順は052-13の順でしたいんだろうけど、子の若い方に行っちゃうから013-52っていう行きがけ順になっちゃうね。そういうのは困るから青い方を作るようにしようってことで、最も左の木の根となるように、みたいな定義になるんだね。きっと
- で、あまり重要じゃないけど $L > R$ となる配列はいらないので塗りつぶします



- 初期化と知らない部分隠すのが終わったね
- いよいよ森の配列を作りに行く。

• $DP_{forest}[L][R]$ = 以下のsum

1. $DP_{tree}[L][i-1] * DP_{forest}[i][R]$ ($L < i \leq R$ かつ $C_L < C_i$)

2. $DP_{tree}[L][R]$

◦ $DP[0][0]$ を考える

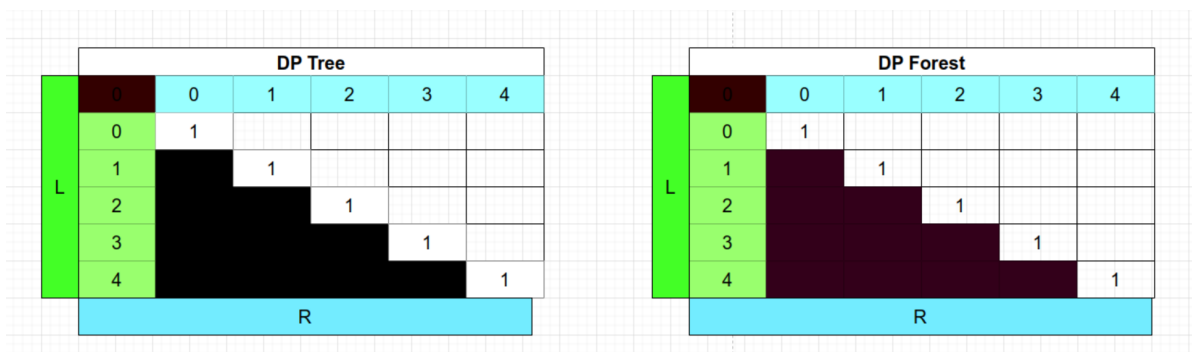
- 実際、定義から考えると、頂点1のみからなる森なので、1通りしかないんだけど、遷移式通りにやります。
- 遷移①を満たす i ($L < i \leq R$) が一つも存在しないので、①は0通りです
- 遷移②は、DPTreeの同じ場所を見てやればいいので、1通りです
- よって、0+1で1通りです。

◦ $DP[1][1]$ を考える、 $DP[2][2]$ を... $DP[4][4]$ を考える

- $DP[0][0]$ のときと同じで条件を満たす i が無いので、 $DP_{tree}[i][i]$ からの遷移しか生まれません

◦ 表に反映します。

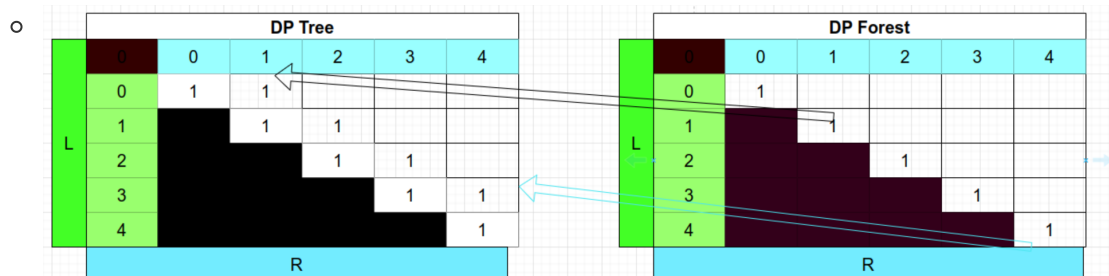
◦ あとついでに、文字サイズ小さかったので大きくします。



- 次にどこの遷移するんだ？って感じなんですけど、

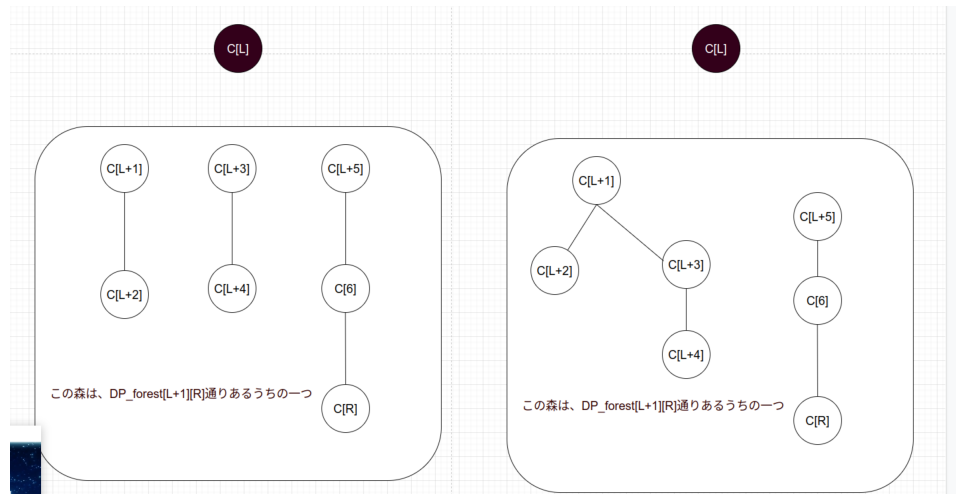
• 遷移1つ目 $DP_{tree}[L][R] = DP_{forest}[L+1][R]$ をします

- 森の方の配列でさっき決めた5つの値があるじゃないですか。木の配列で、その位置の1個上に同じ値を書いてやります。[0][0]の上はないので、4つ値が更新されます。

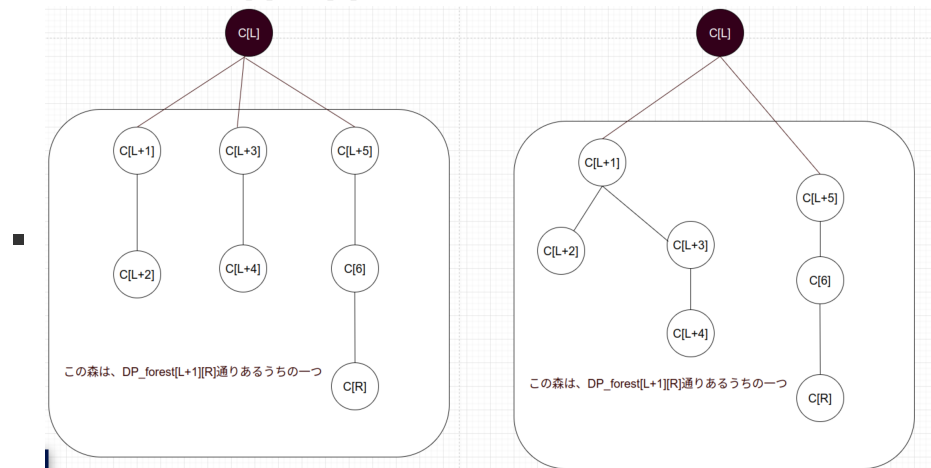


- こうっすね。

- 遷移だけ書いて意味を書かないのはカスなので、意味を説明します。
- $DP_{tree}[L][R] = DP_{forest}[L+1][R]$
 - 森に根をくっつける
 - Lを根として、L+1からRで出来ている森と根をつなぎます。
 - L+1で出来ている森に木が何個あるか知りませんが、3個の木があったとして、それぞれの木の根とLをつないでやります。
 - このとき、森は $DP_{forest}[L+1][R]$ 通りありますが、各森に対して、新しい根(L)の付け方は一通りしかないので、 $DP_{tree}[L][R] = DP_{forest}[L+1][R]$ というわけだ。
 - 補足すると、森は $DP_{forest}[L+1][R]$ 個あるわけだが、全部の森が、行きがけ順にしたときに条件を満たすようになっている
 - 図にすると



- こんな感じで、森が2つあったとき、C[L]を森とつなげることを考える
- 考えると言っても、C[L] (黒い頂点)と、木、の根をつなぐしか無いので、どちらの場合のC[L] ... C[R]からなる木も1通りにしかできません。



- DPテーブルを作る続きに戻ります。
 - DPテーブルどんな感じだったっけ？

		DP Tree					
L	R	0	1	2	3	4	
	0	1	1				
	1		1	1			
	2			1	1		
	3				1	1	
	4						1

		DP Forest					
L	R	0	1	2	3	4	
	0	1					
	1		1				
	2			1			
	3				1		
	4						1

- こんな感じです。
- 次は森のテーブルです。
- なんとなくテーブル埋める順番もわかってきたかと思います。
- 森のテーブル1個埋めたら、Treeのテーブルも1個埋めれるんだなっていう感じですね
- 次は、 $DP_{forest}[0][1]$ を埋めます
 - $C = \{1\ 5\ 3\ 2\ 4\}$
 - 遷移式
 - $DP_{forest}[L][R] =$ 以下の sum
 1. $DP_{tree}[L][i-1] * DP_{forest}[i][R] (L < i \leq R \text{かつ} C_L < C_i)$
 2. $DP_{tree}[L][R]$
 - 2つ目の方は、Treeの配列の同じ位置を見ればいだけなので「1」ですね
 - 1つ目のほうが面倒くさい。
 - $DP_{tree}[0][i-1] * DP_{forest}[i][1]$
 - $L < i \leq R$ を満たす i は1しかない。
 - $C_L = 1 < 5 = C_i$ なので条件満たしてますね。よって $i=1$ はokだということが分かります。
 - $DP_{tree}[0][0] * DP_{forest}[1][1] = 1$
 - sum とって2っすね

	0	1	2	3	4
0	1	1			
1		1	1		
2			1	1	
3				1	1
4					1

	0	1	2	3	4
0	1	2			
1		1			
2			1		
3				1	
4					1

- 意味を考えます。
 - 2つ目の遷移式に関しては、木も森だからってだけ。
 - 1つ目の遷移式の意味は、「森の左に木をくっつける」とあります。
 - 左にくっつける、という表現がちょっとわかりにくいので僕の表現にします
 - $DP_{forest}[i][R]$ 通りある森を一つ選んだとき、その左に $DP_{tree}[L][i-1]$ 通りある木をおく
 - ちょっとわかりにくかったら、 $C_i \sim C_R$ で作れる森の左に $C_L \sim C_{i-1}$ で作れる森をおくとか？
 - 注意点としては、 $C_L < C_i$ じゃないと C_L を根とする木を、森の中で最も左にできないことですね。
 - 掛け算になっている理由ですが、森が X 通り、木が Y 通り考えられるとき、木をどの森の最左に置くかを考えます
 - すると、森1通りに対して、木の置き方は「最も左におく」という1通りしか考えられません
 - それぞれの木に対して、 X 個ある森のうちどれを選ぶか、ってことで、 X 通り
 - 木も Y 個あるから、木と森の組み合わせで $X \times Y$ 通りってわけだ。
- $DP_{forest}[1][2]$ を埋めます
 - $C = \{1\ 5\ 3\ 2\ 4\}$
 - 遷移式
 - $DP_{forest}[L][R] =$ 以下の sum

1. $DP_{tree}[L][i-1] * DP_{forest}[i][R] (L < i \leq R \text{かつ} C_L < C_i)$

2. $DP_{tree}[L][R]$

○ 1つ目のほうが面倒くさい。

○ $DP_{tree}[1][i-1] * DP_{forest}[i][2]$

■ $1 = L < i \leq R = 2$ を満たす*i*は2しかない。

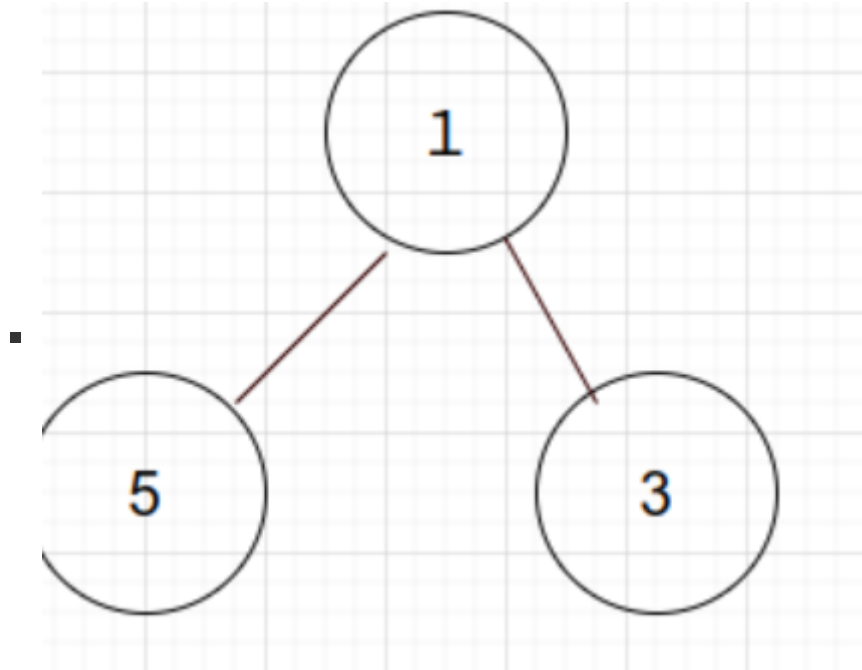
■ $C_L = 5 > 3 = C_i$ なので条件を満たしていません。

■ $DP_{tree}[0][0] * DP_{forest}[1][1] = 1$ は使っちゃいけません。

■ これは、下図のような森は認められないワってことを言っています



■ 実際、こういう森を作ってしまった場合、根をつけると次のようになりますね



■ こうすると、行きがけ順が1->3->5となってしまう、 $C=\{1,5,3,2,4\}$ になっていないことが分かります

■ こういうのを排除するための条件式だったんですね

○ 2つ目の遷移式に関しては、木のほうのDP見て、1なので、sumは1です。

■

■

		DP Tree					
L		0	1	2	3	4	
	0	1	1				R
	1		1	1			
	2			1	1		
	3				1	1	
	4					1	

		DP Forest					
L		0	1	2	3	4	
	0	1	2				R
	1		1	1			
	2			1			
	3				1		
	4					1	

- 森の配列が1個うまると、木の配列も1個埋めれます。

$DP_{tree}[L][R] = DP_{forest}[L+1][R]$ です。

		0	1	2	3	4
L	0					
	0		1	1	1	
	1			1	1	
	2				1	1
	3					1
	4					1
	R					

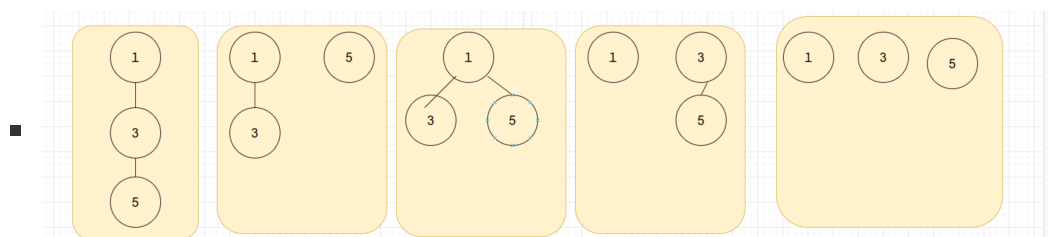
		0	1	2	3	4
L	0					
	0		1	2		
	1			1	1	
	2				1	1
	3					1
	4					1
	R					

- って感じで、残りも埋めます。

		0	1	2	3	4
L	0					
	0		1	1	1	
	1			1	1	
	2				1	1
	3					1
	4					1
	R					

		0	1	2	3	4
L	0					
	0		1	2		
	1			1	1	
	2				1	1
	3					1
	4					1
	R					

- ホイ。
- ここまでで、区間の幅が1のところまで埋め終わりました。
- $DP_{forest}[0][2]$ を埋める
 - $C = \{1\ 5\ 3\ 2\ 4\}$
 - 遷移式
 - $DP_{forest}[L][R] =$ 以下の sum
 - $DP_{tree}[L][i-1] * DP_{forest}[i][R] (L < i \leq R \text{ かつ } C_L < C_i)$
 - $DP_{tree}[L][R]$
 - ①の式から
 - $DP_{tree}[0][i-1] * DP_{forest}[i][2]$
 - $0 = L < i \leq R = 2$ を満たす*i*は1,2の2つ
 - *i*=1はok?
 - $C_L = 1 < 3 = C_i$ なのでok
 - *i*=2はok?
 - $C_L = 1 < 5 = C_i$ なのでok
 - で、あと②の式も足して 多分5
 - $C[0] \sim C[2]$ で作れる森、見てみますか。



- 確かに5通りだ！！

