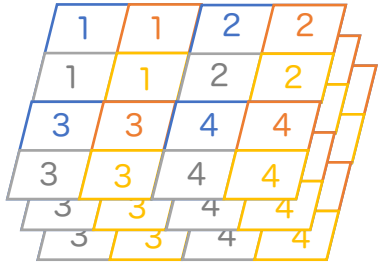
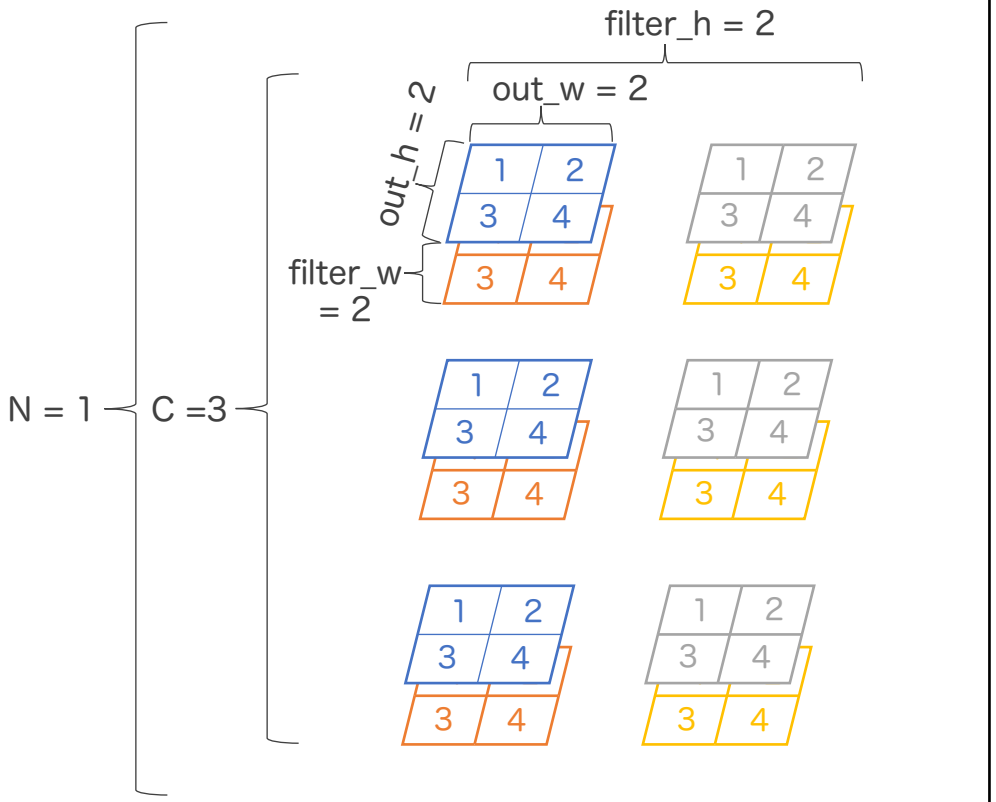


## ① im2colの入力画像（今回は3チャンネル，縦4ピクセル，横4ピクセルを仮定）

- 今回は，フィルタサイズ2x2、ストライド2，パディングサイズ0のケースを仮定
- img[:, :, y:y_max:stride, x:x_max:stride]で参照される領域ごとに色を割り当てている
  （=DAY4スライド p.78の「畳み込みフィルタの〇〇の部分」ごとに色を割り当てている）

## ② 二重ループ終了直後のcol

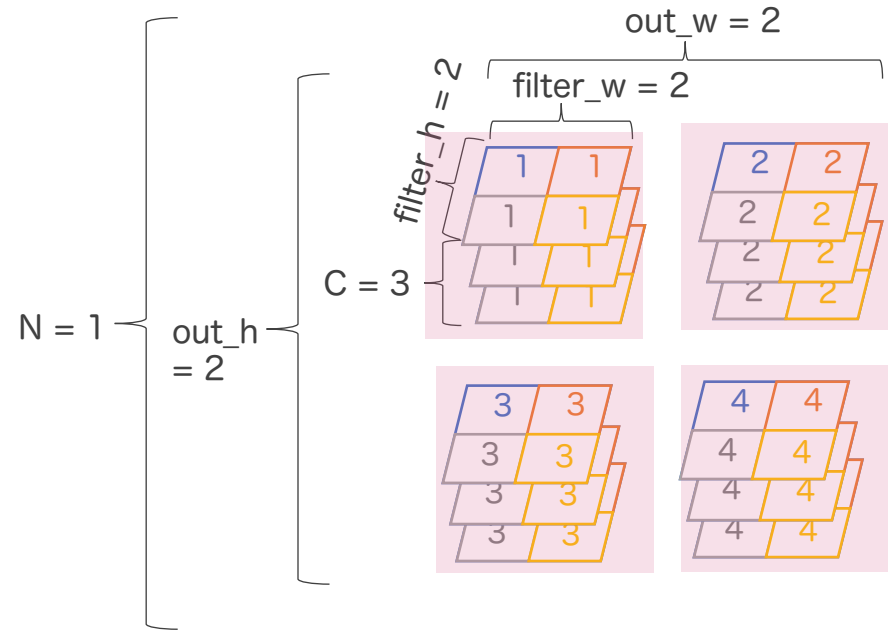
colのshape: (N, C, filter_h, filter_w, out_h, out_w)




## ③ col.transpose(0, 4, 5, 1, 2, 3)の直後

col.transpose(0, 4, 5, 1, 2, 3)のshape:
(N, out_h, out_w, C, filter_h, filter_w)

で塗られた3次元テンソル1つ1つがフィルタと積和を取る部分！これらをベクトルにして並べれば完成！




④へ

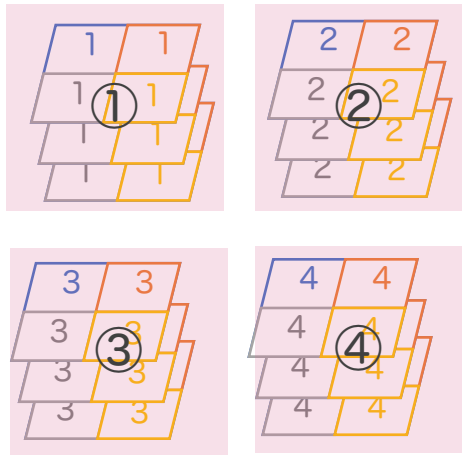
④ col.transpose(0, 4, 5, 1, 2, 3).reshape(N * out_h * out_w, -1)の直後
col.transpose(0, 4, 5, 1, 2, 3).reshape(N * out_h * out_w, -1)のshape:
(N * out_h *out_w, C * filter_h * filter_w)

C * filter_h * filter_w = 3 * 2 * 2 = 12

N * out_h * out_w
= 1 * 2 * 2
= 4

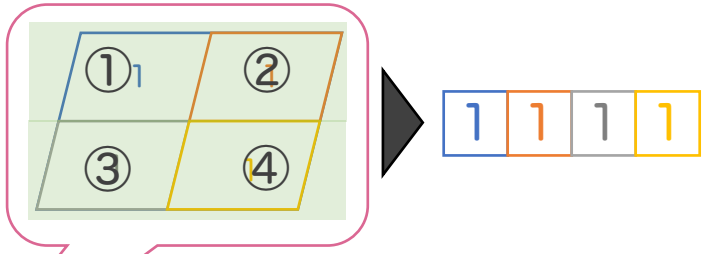
すべて1列に並べたあとに
(N * out_h *out_w, C * filter_h * filter_w)
の配列に，並べた列の左から順に，
左上から値を詰めていく

reshapeはまずすべてのデータを一列に並べることを考える．データはインデックスが若い順に並べていく．まずは．3次元テンソルを見る順番は①→②→③→④の順

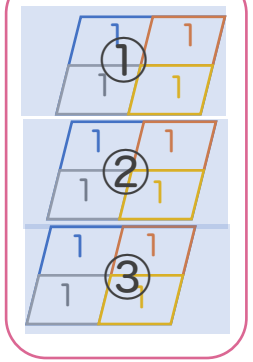


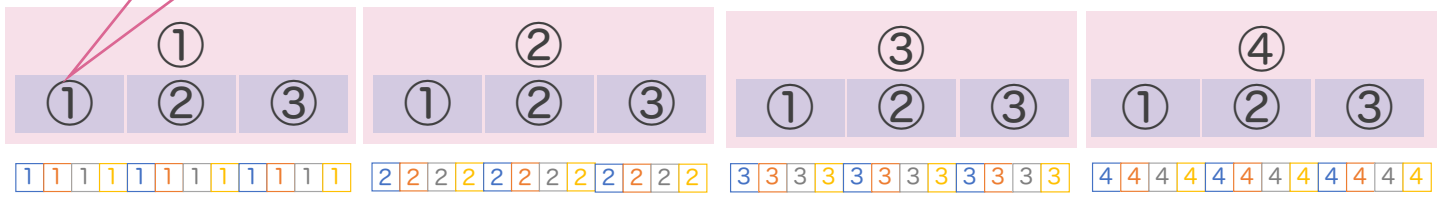
これらを一列に並べるので以下となる

インデックスが若い順に見る



チャンネル方向も同様にインデックスが若い順に見る





実際に出力を確認したいときは,
DAY4「4_4_im2col_col2im.ipynb」で
以下のようにすればよい

```
# データ数が1の場合の確認
x1 = np.ones((1,3,4,4))
tmp = np.array([[1, 1, 2, 2],
                [1, 1, 2, 2],
                [3, 3, 4, 4],
                [3, 3, 4, 4]])
x1[:, 0] = tmp.copy()
x1[:, 1] = tmp.copy()
x1[:, 2] = tmp.copy()

# x1 = np.random.randn(1,3,4,4)
print("x1=",x1.round(2))
print()
print("パディングなし")
col1 = im2col(x1, 2, 2, stride=2, pad=0)
print("col1=",col1.round(2))
```

実行結果

```
x1= [[[[1. 1. 2. 2.]
   [1. 1. 2. 2.]
   [3. 3. 4. 4.]
   [3. 3. 4. 4.]]

  [[1. 1. 2. 2.]
   [1. 1. 2. 2.]
   [3. 3. 4. 4.]
   [3. 3. 4. 4.]]

  [[1. 1. 2. 2.]
   [1. 1. 2. 2.]
   [3. 3. 4. 4.]
   [3. 3. 4. 4.]]]]

パディングなし
col1= [[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2.]
 [3. 3. 3. 3. 3. 3. 3. 3. 3. 3. 3. 3.]
 [4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4. 4.]]
```