参考） 2038年問題

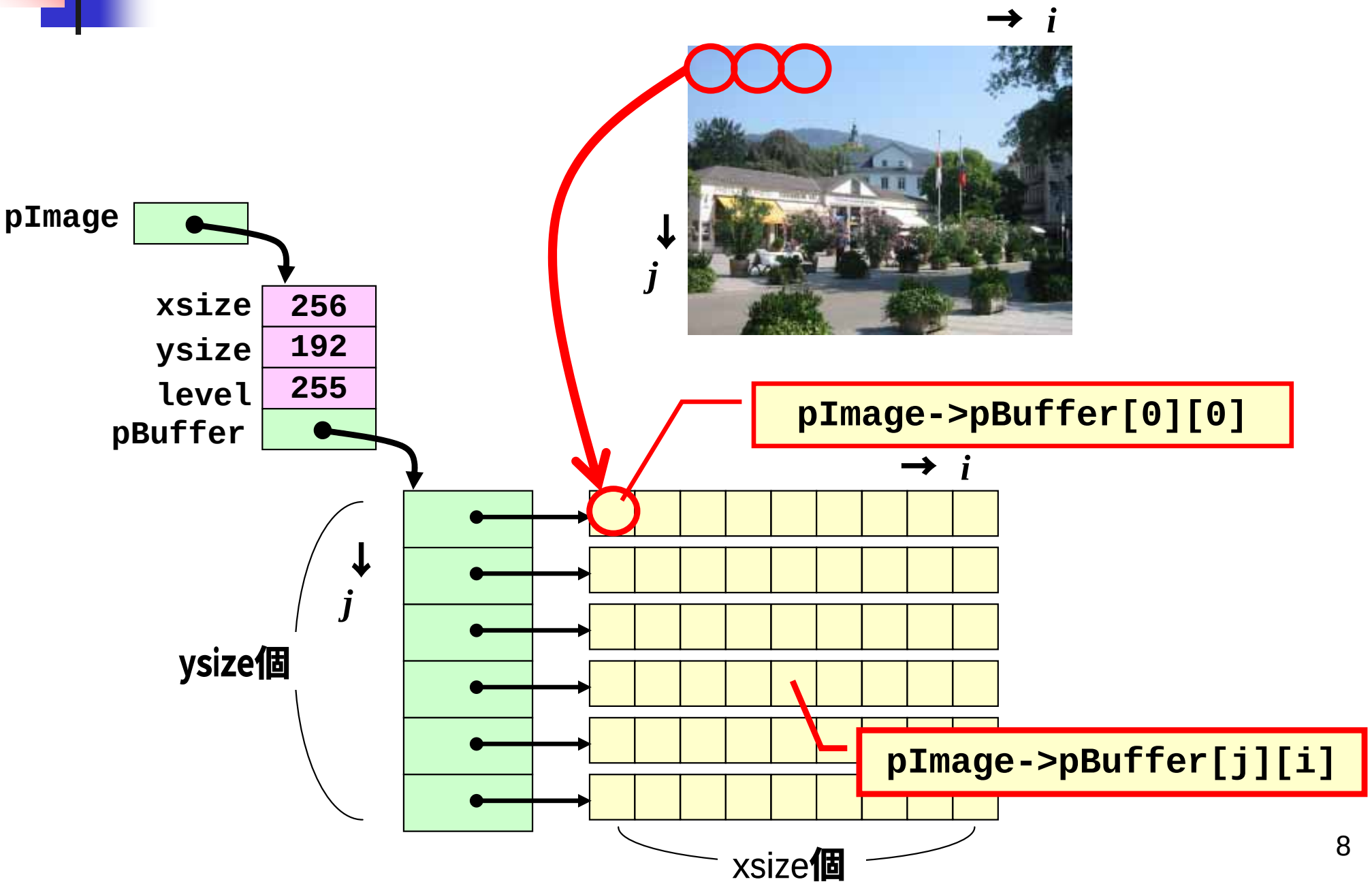
- グリニッジ標準時(GMT)の2038年1月某日を過ぎるとシステムが正しく時刻を認識できなくなる問題
- なぜ起きる？
  - システム内の時刻：GMT1970年1月1日0時0分0秒からの経過秒数
  - 時刻データに4バイトの符号付き整数を使用(最大値2^{31}-1)

  → 経過秒数が最大値を超える

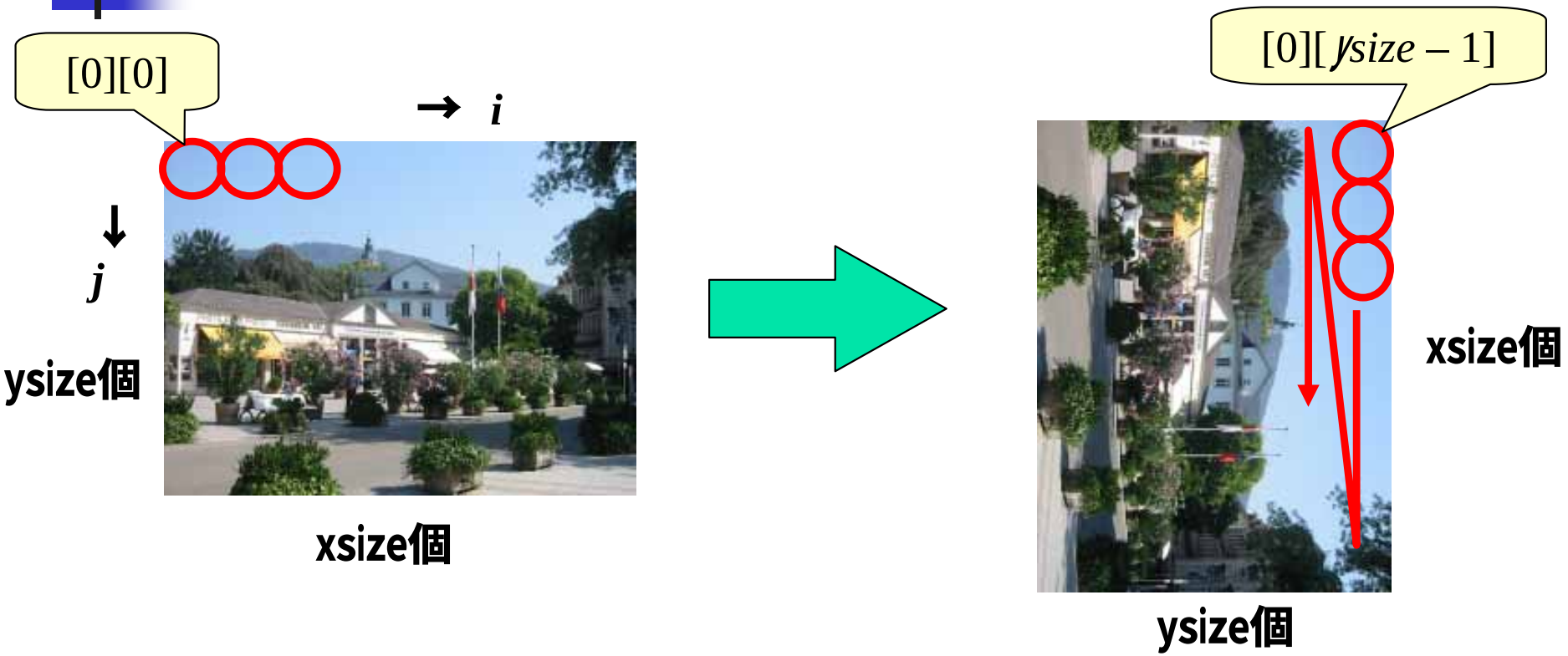
- トラブル例： 2004年1月11日の銀行ATMのトラブル
  (2038年までのちょうど半分)

変数の型や表現できる値の範囲に
注意することがシステムを作る上で重要

画像バッファの様子


→ i
j
pImage
xsize 256
ysize 192
level 255
pBuffer
pImage->pBuffer[0][0]
→ i
j
ysize個
pImage->pBuffer[j][i]
xsize個

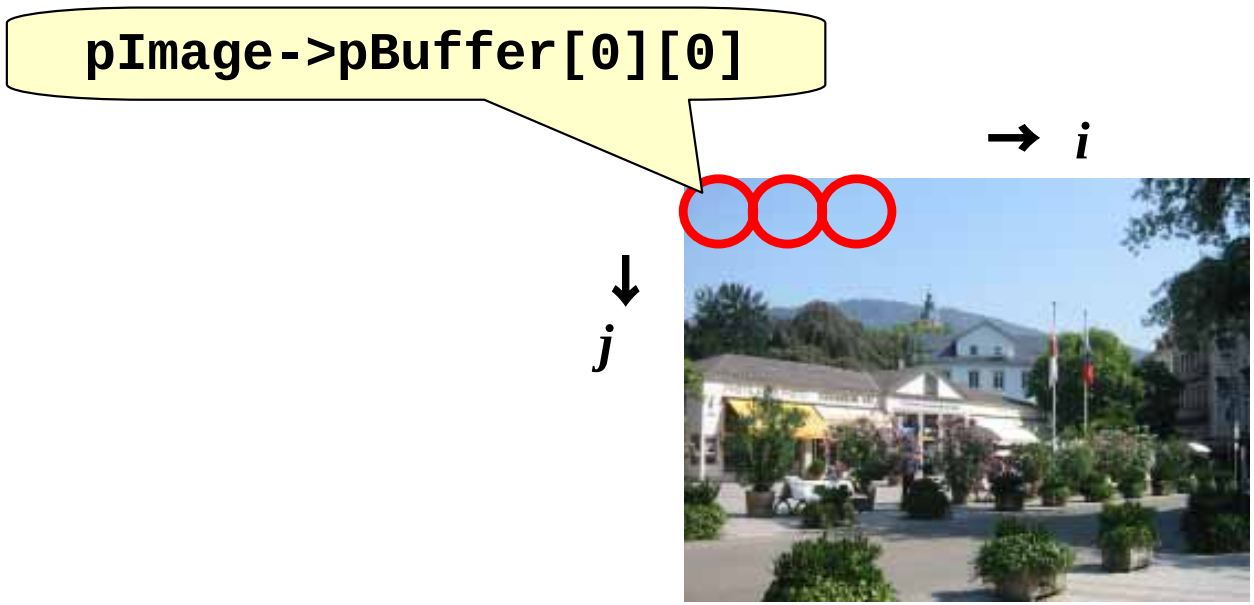
画像の90度回転　ipRotateImage



関数内で作業用の画像バッファメモリを確保し，順に画素を埋めていけばよい．

iioMallocImageBuffer
iioFreeImageBuffer

画素のビットマスク ipBitMask



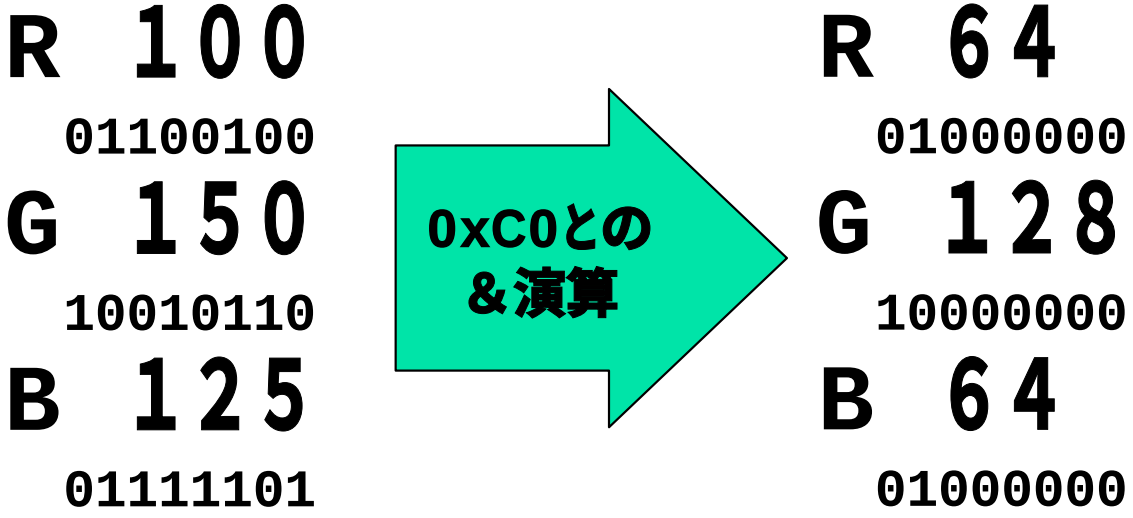
1画素ずつ処理していく
1画素はR,G,Bから成る構造体で表現している
→ R,G,Bのそれぞれに対して処理を行う

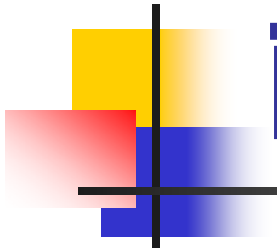
画素のビットマスク ipBitMask

```
pImage->pBuffer[j][i].r &= 0xC0;
pImage->pBuffer[j][i].g &= 0xC0;
pImage->pBuffer[j][i].b &= 0xC0;
```

2進表現で
1100 0000

下位6ビットが0なので
ビットごとのAND演算
をすると，下位6ビット
が0にマスクされる．





画素のビットマスク ipBitMask

```
pImage->pBuffer[j][i].r &= 0xC0;
pImage->pBuffer[j][i].g &= 0xC0;
pImage->pBuffer[j][i].b &= 0xC0;
```

サンプルプログラム3：
各画素の下位”6”ビットを0でマスク→0xC0との＆演算

演習課題：
各画素の下位”4”ビットを0でマスク →何を使えばよいか？