コンピュータ・サイエンス1

第13回
データ圧縮

人間科学科コミュニケーション専攻
白銀 純子

第13回の内容

- データ圧縮

設問1

- 下記の中で正しい説明をすべて選びなさい。

1. 音の標本化では、標本化の間隔により、音の強さをどのくらい細かく取り込むかのレベルが決定される
   - 標本化により、音の高さのレベルが決まる
2. 音楽CDと固定電話で同じ品質で音声を標本化・量子化している
   - 固定電話の方が音楽CDよりも低い品質で標本化・量子化している
3. 標本化と量子化を経て取り込まれたデータは、もとのデータそのままにはなっていない
   - 量子化により、アナログ信号のグラフ上にはない点を取り込んでいるので、なっていない
4. コンピュータのディスプレイで表現されている色は、シアン・マゼンタ・黄色・黒の4つの光に濃淡をつけて混ぜ合わせている
   - ディスプレイの色は、赤・緑・青の3つの光に濃淡をつけて混ぜ合わせている

解答: 3

設問2

- 比較(標本化)(1)において、画像1と画像2のどちらが小さく表示されたかを回答し、それはなぜかを考察しなさい。

解答: 小さく表示されたのは画像1のはず

- **理由: 画像1の方が解像度が小さいので、画像の縦・横の点の数が少ないため**

設問3

- Gimpで画像1の解像度を大きくし、画像2と比較したとき、画像の品質が同じであったかを回答し、その理由を考察しなさい
  - 同じであったか違ったか
  - 同じであったら同じであった理由、違っていれば違った理由

解答: 違ったはず

- **理由: 画像1の方が解像度が小さいので、解像度を大きくすると、点の大きさが大きくなり、いびつな画像になるため**

前回の質問の解答

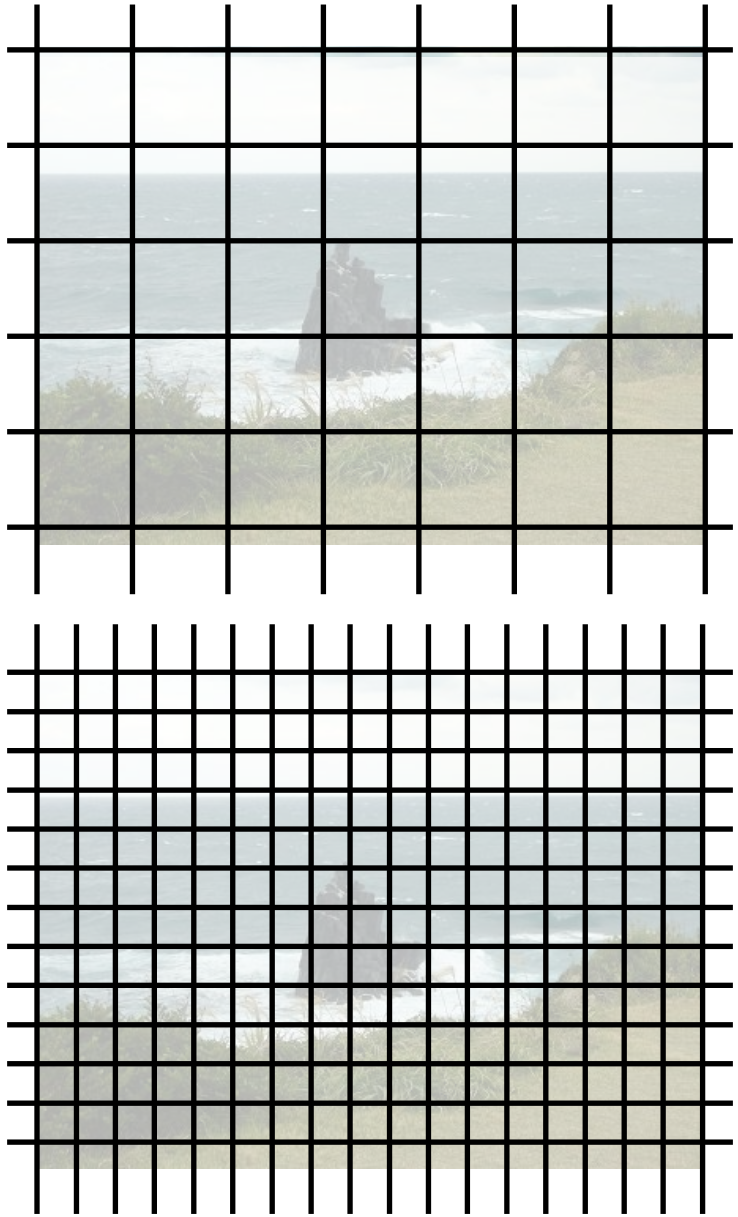
解像度[1]

- スキャナでの解像度と、コンピュータ上での解像度は、少し違う意味

- スキャナでの解像度: 1インチあたりをいくつの点として区切って
  コンピュータに取り込むか(= dpi)
  - dpi (Dot Per Inch): スキャナでの解像度の単位
    - 300dpi: 1インチあたりを300個の点として区切ってコンピュータに取り込む
    - 50dpi: 1インチあたりを50この点として区切ってコンピュータに取り込む
      - dpiが大きければ、コンピュータに取り込んだときの画像を表す点の数が多い
      - dpiが小さければ、コンピュータに取り込んだときの画像を表す点の数が少ない

解像度[スキャナ]

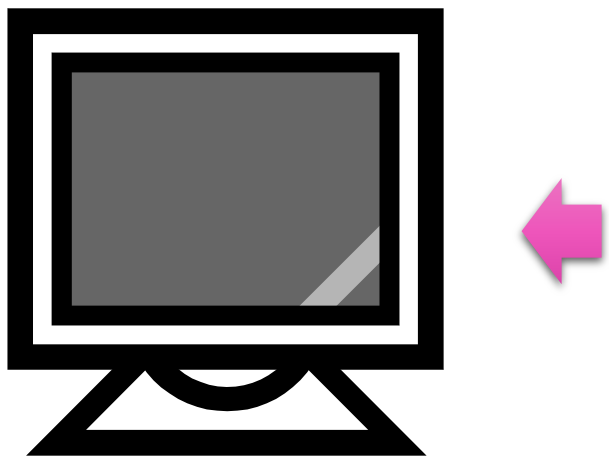
解像度が小さい(dpiが小さい)
=1インチあたりの点が少ない



解像度が大きい(dpiが大きい)
=1インチあたりの点が多い

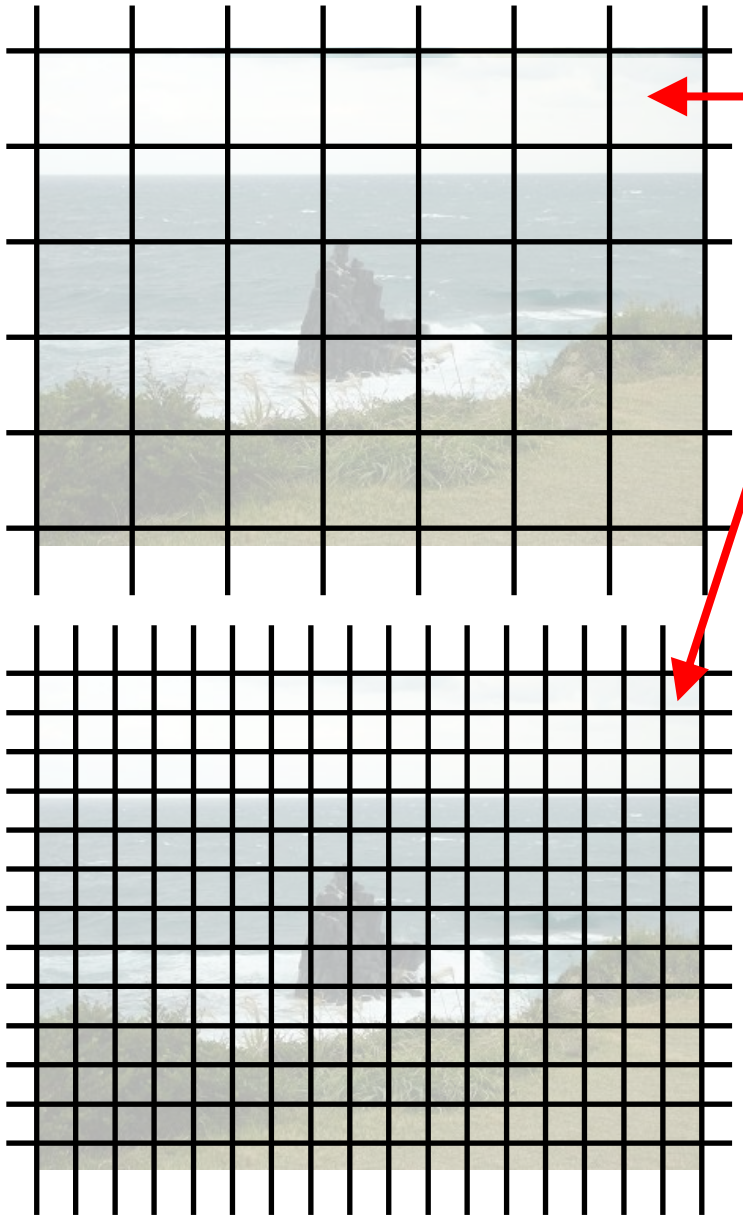
解像度[コンピュータ]

- コンピュータでの解像度: ディスプレイや画像を、縦横いくつの点で表すか
  - 「横の点の数×縦の点の数」という形で表現(単位: ドットまたはピクセル)
    - Ex. 1920×1200ドット: 横1920個、縦1200個の点で表示(12インチのノートPCなど)



- 肉眼では見えないくらいの小さな点が縦横にびっしり並んでいる
- 1つ1つの点の大きさは全て同じで正方形をしている
- 1つ1つの点を様々な色で光らせることにより表示している
  - ✓ 赤・緑・青の3つの光に濃淡をつけて混ぜ合わせて色を作っている
  - ✓ 人間にとって意味のあるないように見せている

解像度[スキャナで取り込んだ画像]



- スキャナで画像を取り込むときは、このように区切ったマス目1つ1つを点としてコンピュータに取り込み
  - ✓ Ex. 写真(L版, 横127mm・縦89mm)の場合
    - 300dpiで取り込むと: 1500×1051ドット
      - 横: (127mm÷25.4mm)×300≒1500
      - 縦: (89mm÷25.4mm)×300≒1051
    - 50dpiで取り込むと: 250×175ドット
      - 横: (127mm÷25.4mm)×50≒250
      - 縦: (89mm÷25.4mm)×50≒175
- コンピュータ上で表示する点の大きさはすべて同じ
  - ✓ つまり、ディスプレイでの点の大きさ

- dpiを小さくすると、コンピュータ上での表示は小さくなる
- dpiを大きくすると、コンピュータ上での表示は大きくなる

Question!

データ圧縮と情報量

画像データの伝送[1](p. 25)

- 地上デジタル放送: 1440×1080の解像度でフレームレートが約30fps
  - fps(frame per second): 1秒あたりに切り替える画像の数

動画の世界では「フレーム」と呼ぶ

Ex. 1つの点を24ビット(3byte)で表現すると…
(1440×1080画素)×(3byte)×(30フレーム/秒)
=139968000byte/秒
≒140Mbyte/秒
(ただし、音声なし)

画像データの伝送[2](p. 25)

- デジタル放送: 140Mバイト/秒の伝送が必要
  - テレビ局から家庭に映像を送るときに、1秒間に140Mバイト送れないと、なめらかな映像が見られない
  - Ex. 家庭のコンピュータのネットワーク環境
    - ADSL: 50Mビット/秒(bps) = 6.25Mバイト/秒
    - 光: 100Mビット/秒(bps) = 12.5Mバイト/秒

理論上の速さで、実際はもっと遅い

140Mバイト/秒は莫大な伝送量!!

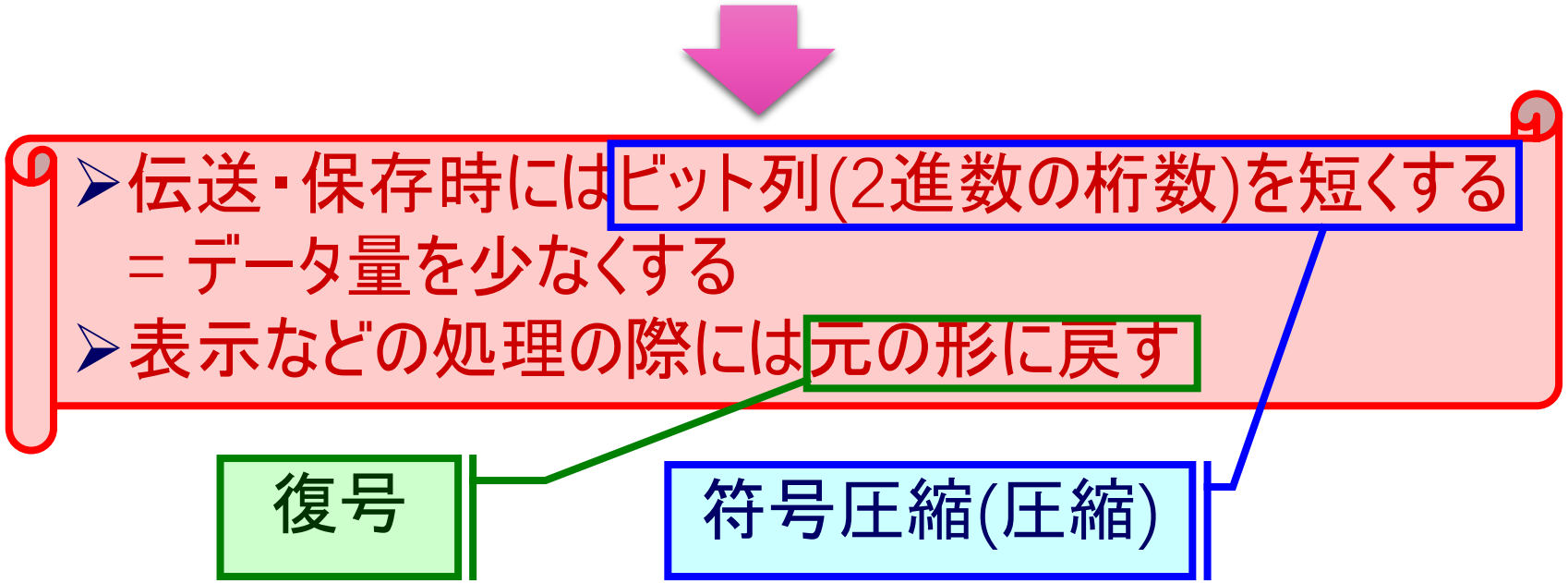
画像のデジタル表現[4](p. 25)

- 140Mバイト/秒は現実的に伝送不可能
  →データを圧縮して伝送
  - 圧縮: 決まった手順に従ってデータのサイズを小さくすること
- 静止画
  - 画像の中の特定の領域の色の変化は少ない
- 動画
  - 動画の中で実際に動く部分は大きくない
  - 動く部分を過去の様子から大体予測できる

デジタル放送では、この性質を使って圧縮
(圧縮技術: MPEG-2)

圧縮と復号(p. 26)

- 数値・文字・画像・音声: 符号化(2進数に変換)して処理
- 特に画像・音声はデータ量が多い
  - 伝送に時間が多く必要
  - 保存場所の容量が多く必要



可逆圧縮(p. 26)

可逆圧縮

- 復号時に1ビットの違いもなく元のビット列が復元される圧縮法
  - 圧縮したファイルから、圧縮前のファイルを取り出すことができる
- 非可逆圧縮に比べて、ビット列をあまり短くすることはできない
  - 非可逆圧縮よりもデータ量の減量分は少ない

非可逆圧縮(p. 26)

非可逆圧縮

- 復号時に元のビット列とは多少の違いが生じてしまう圧縮法
  - 圧縮したファイルから、圧縮前のファイルを取り出すことができない
- 可逆圧縮に比べて、ビット列がかなり短くなる
  - 可逆圧縮よりもデータ量の減量分が多い

数字・文字情報(p. 26)

- 復号したときに内容が変わってはならない
  - 文書中の1文字が抜け落ちたら…?
  - 数値の小数点の桁数が少なくなったら…?
- 一般的に(画像・音声に比べて)データ量は少ない
  - 圧縮によるデータの減量分はそれほど(画像・音声ほど)多くなくて良い

可逆圧縮が使われる

画像・音声情報(p. 26)

- 復号したときに内容が多少変わっても良い
  - 画像: 表示したときに見た目が変わらなければ良い
  - 音声: 聞いたときに元と同じように聞こえれば良い
- 一般的に(数値・文字に比べて)データ量が多い
  - 圧縮によるデータの減量分が(数値・文字に比べて)多いことが必要

↓

非可逆圧縮が使われる

- 動画・音声のデータ量は膨大なので、ほとんどの場合非可逆圧縮が使われる
- 静止画のデータ量はそれほど多くないので可逆圧縮・非可逆圧縮のどちらも使われる

非可逆圧縮の圧縮法

- 余分な情報を取り除くことで圧縮
  - 画像の場合、同じ色が集まっているところの色の情報など
  - 音声の場合、人間の耳には聞こえないような音

人間の感覚ではわからないものを削るので、見た目・聞いた感じでは品質は変わらない

- 画像は拡大すると、画質が落ちているのがわかる
- 音声は、良いスピーカーを使うと音質が落ちているのがわかる
- 圧縮率を上げる(ファイルサイズをより小さくする代わりに、取り除くものを多くする)と、質が落ちているのがわかる

可逆圧縮の圧縮法(p. 28)

- ハフマン符号化
- ランレングス符号化
- etc.

ハフマン符号化[1](p. 28)

- ハフマン符号化: データ内に出現する情報を統計的に処理し、ビット列の長さを変えて情報を表現
  - これまで: どの情報も同じ長さのビット列で表現
  - 出現率の高い情報を短いビット列で表現
  - 出現率の低い情報を長いビット列で表現

ハフマン符号化[2](p. 28)

Ex. ある地方の320日の天気

天気	天気である日数	ビット列での表現
雨が降っていない	160日(2日に1日)	0
小雨が降っている	40日(8日に1日)	100
適度な雨が降っている	20日(16日に1日)	1010
やや強い雨が降っている	20日(16日に1日)	1011
非常に強い雨が降っている	20日(16日に1日)	1100
強い雨が降っている	20日(16日に1日)	1101
弱い雪が降っている	20日(16日に1日)	1110
大雪が降っている	20日(16日に1日)	1111

「10100100...」は、何日分のどんな天気？

ハフマン符号化[3](p. 28)

- 「10100100…」
  - 最初のビットが「1」なので、「雨が降っていない」という天気ではない
  - 2ビット目が「0」なので、「非常に強い雨」、「強い雨」、「弱い雪」、「大雪」という天気ではない
  - 3ビット目が「1」なので、「小雨」という天気ではない
  - 4ビット目が「0」なので、「やや強い雨」という天気ではない

1日目の天気は「適度な雨が降っている」

それぞれの情報を同じ長さのビット列にしなくても情報の表現は可能

※情報の種類ごとに割り当てるビット数を変える方式: 可変長符号

ハフマン符号化[4](p. 29)

- 8種類の天気を普通のやり方で現すと…？
  - 1種類を3ビットで表現: 320日では、320×3=960ビット

ハフマン符号化では…

雨が降っていない	160日	0	1ビット×160日
小雨が降っている	40日	100	3ビット×40日
適度な雨が降っている	20日	1010	4ビット×20日
やや強い雨が降っている	20日	1011	4ビット×20日
非常に強い雨が降っている	20日	1100	4ビット×20日
強い雨が降っている	20日	1101	4ビット×20日
弱い雪が降っている	20日	1110	4ビット×20日
大雪が降っている	20日	1111	4ビット×20日

合計: 760ビット

➡ 200ビット少なくなっている

※どれだけ少なくなるかは扱う情報によって違う

ランレングス符号化[1](p. 29)

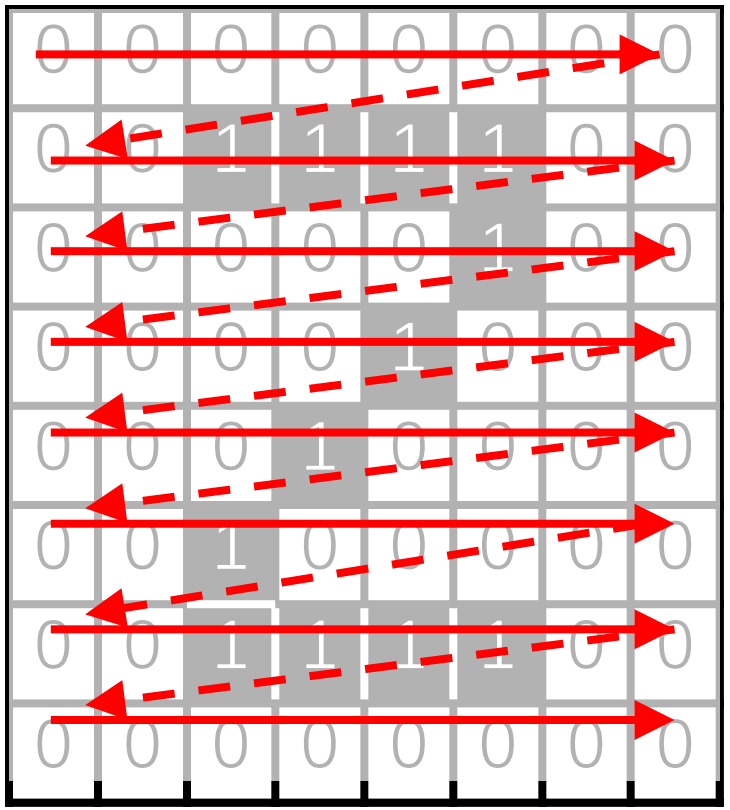
- ランレングス符号化: 白黒2値画像(灰色のない、白と黒のみの画像)の圧縮方法の1つ
  - 主にFAXで使われている

0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0

8×8=64ビット
(2値画像は1つ1つの点を1ビットで表現するため)

ランレングス符号化[2](p. 29)

- 2値画像: 白画素と黒画素はある程度固まっている

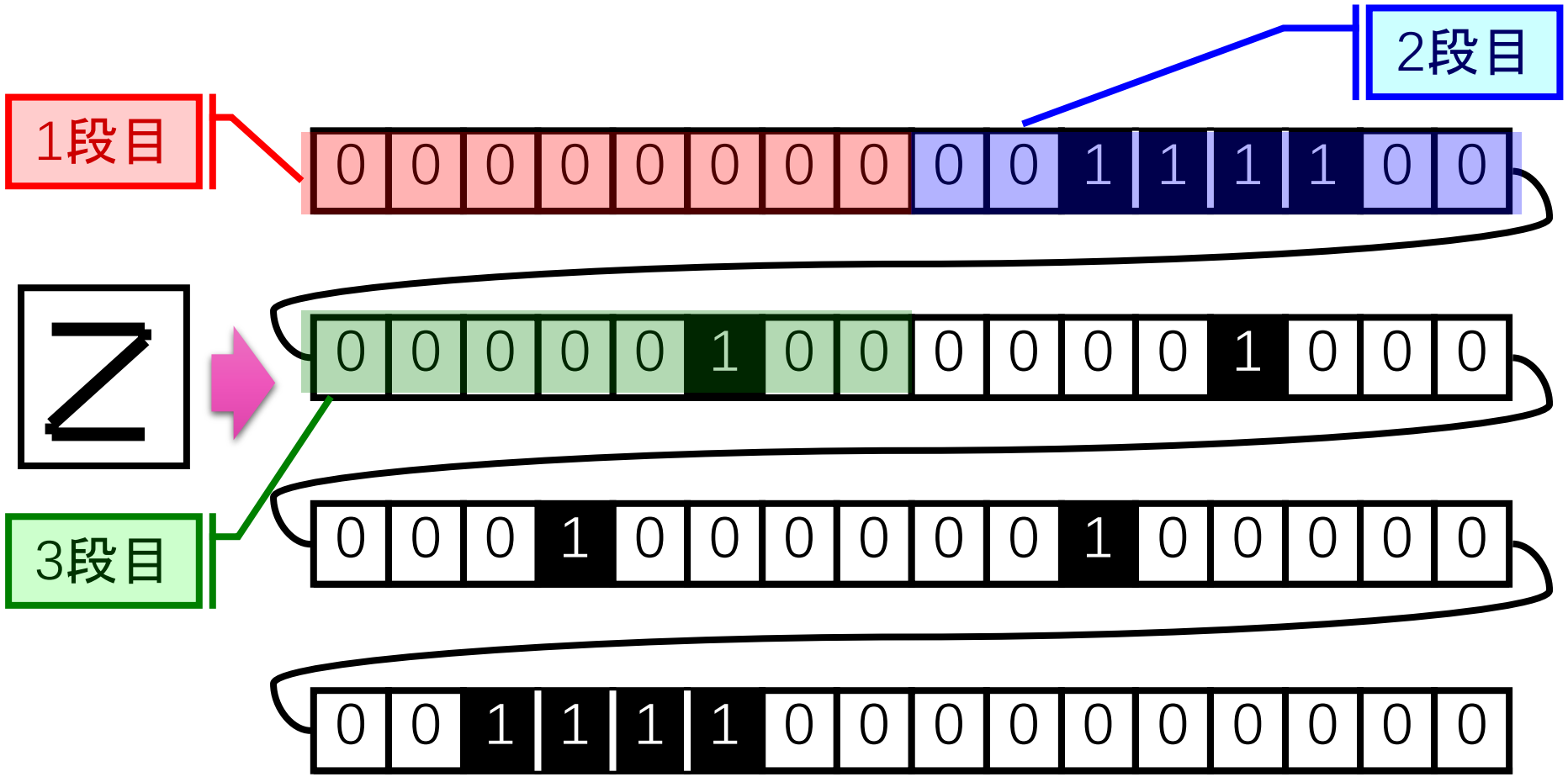


白画素が10個
黒画素が4個
白画素が7個
.....
黒画素が4個
白画素が10個

矢印の方向に画素を見ていったとき

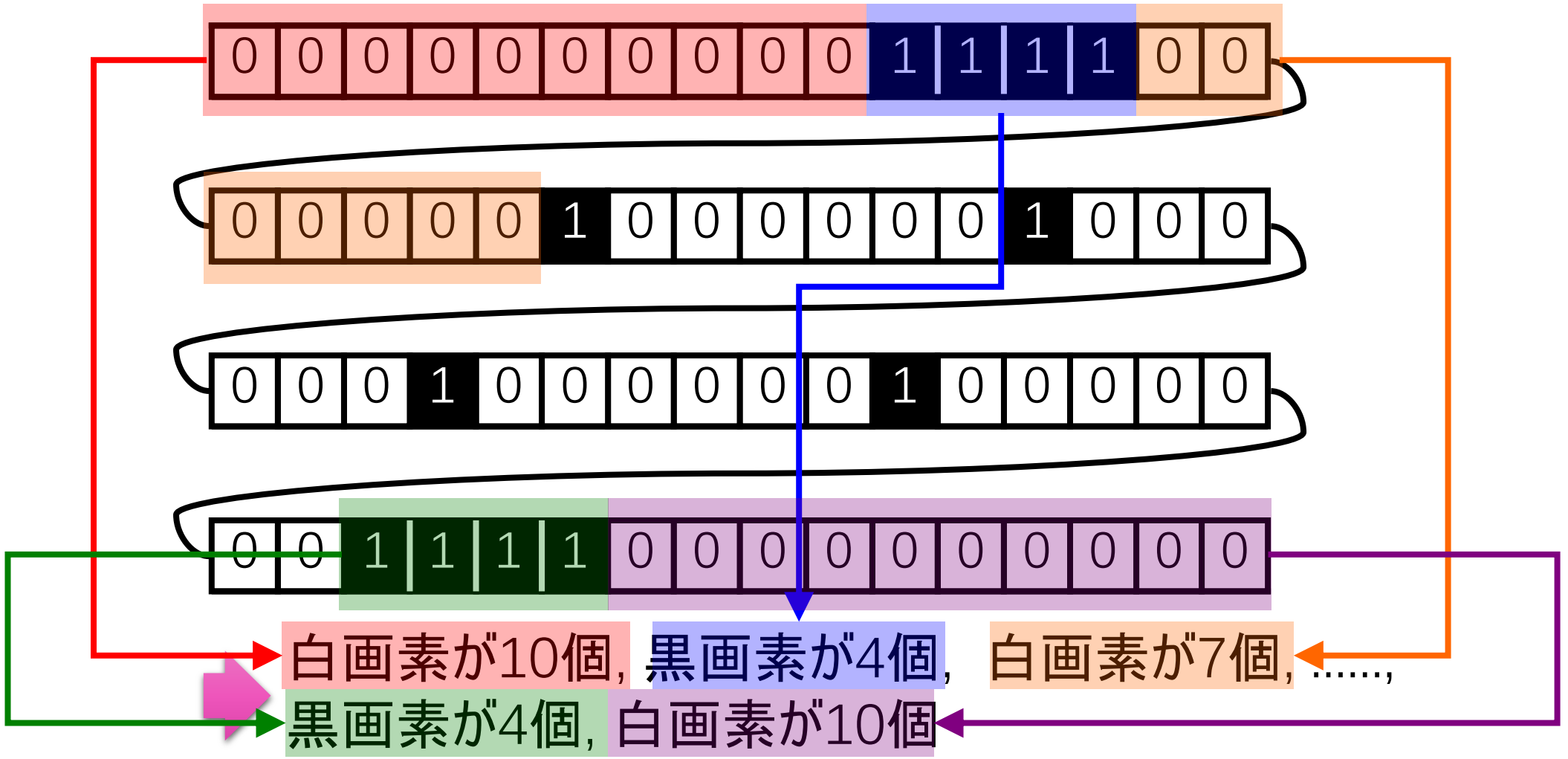
ランレングス符号化[圧縮][1]

- 画像の中の全ての画素を横一列に並べる
  - 横1段目の右に2段目、2段目の右に3段目…とつなげていく



ランレングス符号化[圧縮][2]

- 横一列に並べた画素について、左から順に、連続している白画素と黒画素の数を数えていく



ランレングス符号化[圧縮][3]

白画素が10個　黒画素が4個　白画素が7個　.....
黒画素が4個　白画素が10個

画素の個数を2進数で表すと...

1010　0100　0111　0001　0110　0001　0110
0001　0110　0001　0111　0100　1010

52ビット

※個数の中で「1010」(2進数)が最も大きな個数なので、他の個数も、
2進数で表現したときの桁数を「1010」(4桁)にあわせる

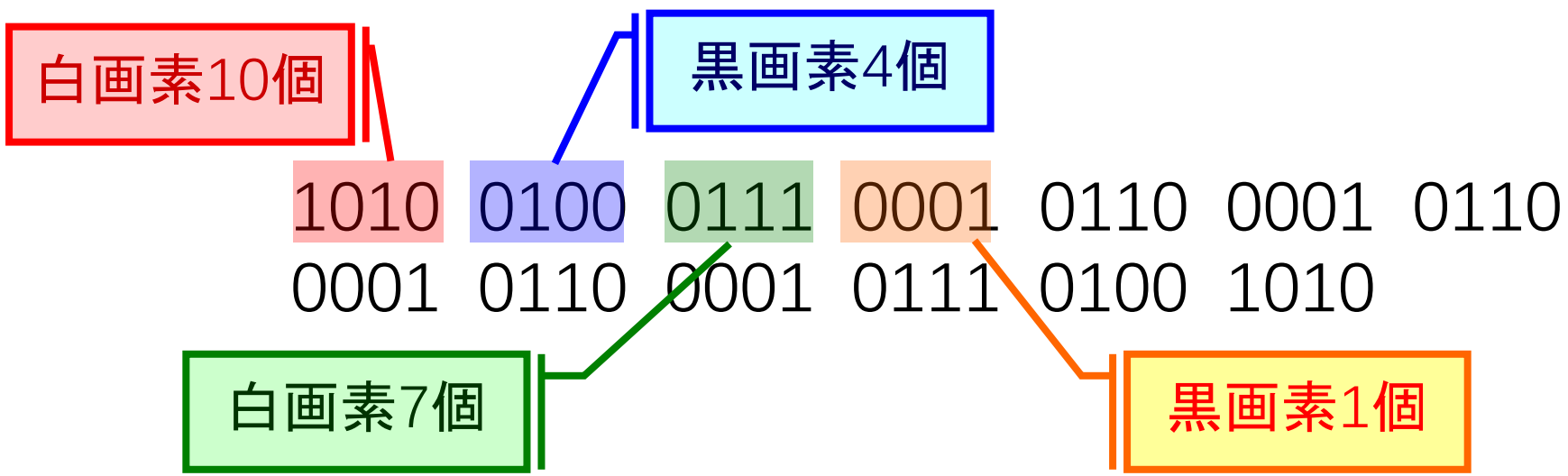
通常の方法で表す(1画素を1ビットで表す)と...
8×8=64ビット

12ビット少なくなっている

※どれだけ少なくなるかは扱う画像の内容によって違う

ランレングス符号化[復号][1]

- 圧縮したものを復号化するには…?
  - 画像の画素の数だけを表したもの(色の情報はない)
  - 白画素の並びと黒画素の並びは必ず交互になる
  - 画像の最も左上隅の画素は白であることが多い
    - 先頭の個数は、白画素の個数として扱う
    - 最も左隅の画素が黒の場合、圧縮時に、最初に出現する白画素の個数を0個としておく

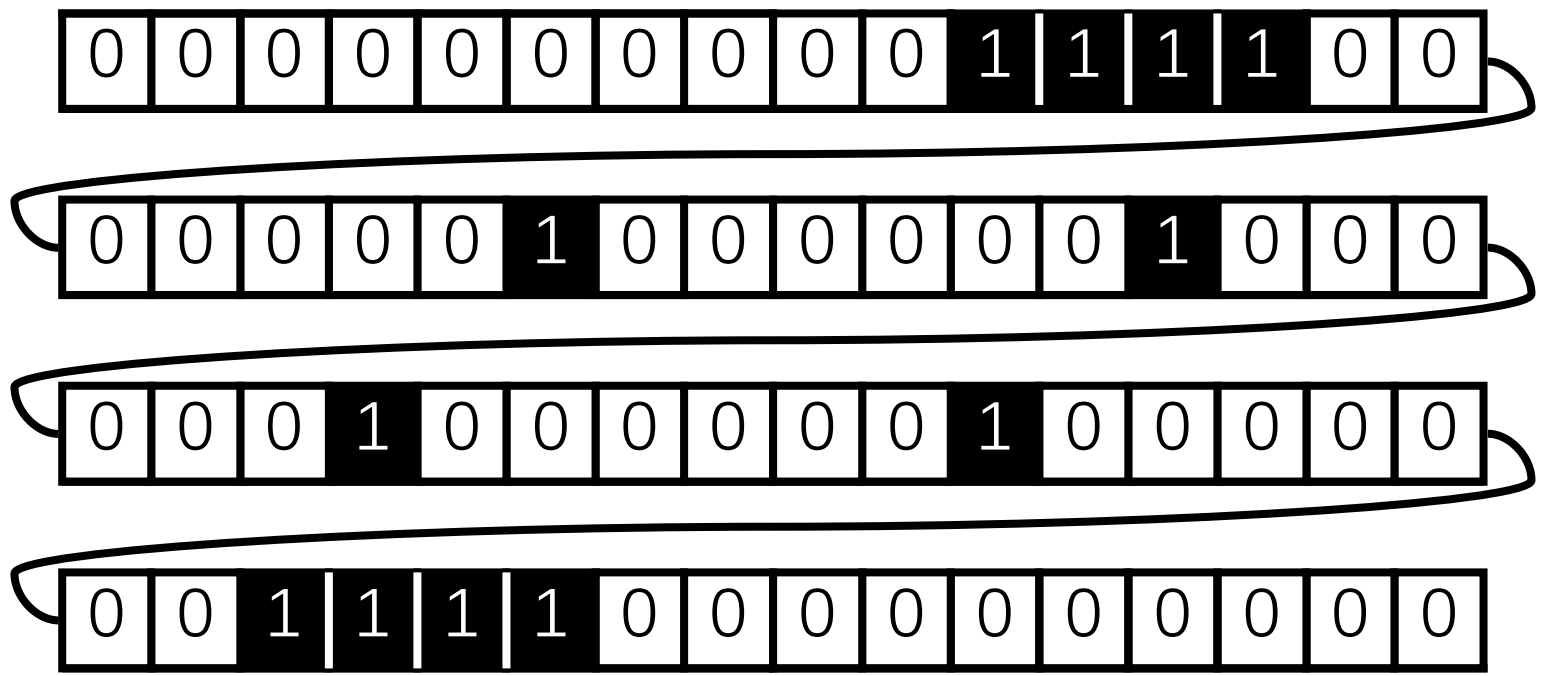


ランレングス符号化[復号][2]

- 画素の個数から、横一列の画素の並びが復元

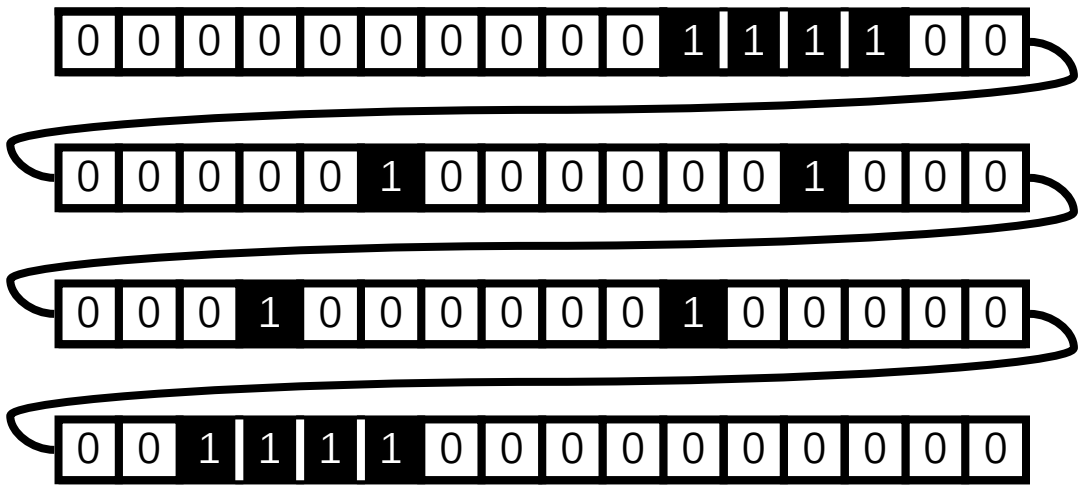
1010 0100 0111 0001 0110 0001 0110
0001 0110 0001 0111 0100 1010

白画素が10個, 黒画素が4個, 白画素が7個, ……,
黒画素が4個, 白画素が10個



ランレングス符号化[復号][3]

- もとの画像の縦横の画素の数は記録されてある
  - 横一列の画素の並びが復元できると、もとの画像も復元できる

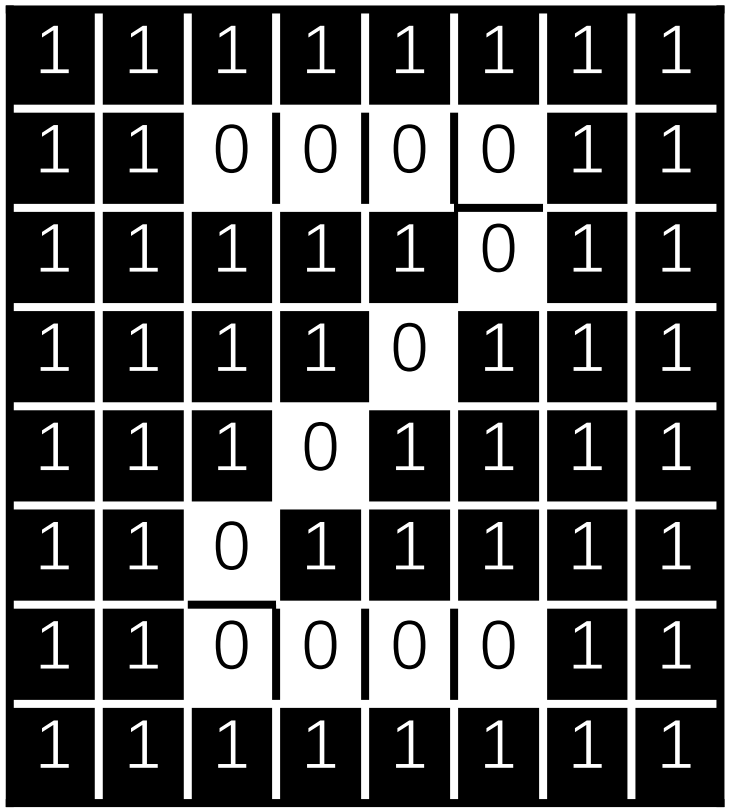


もとの画像は横8個、縦8個の画素

0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0

先頭が黒画素のとき[1]

- 前のスライドの画像の、白と黒を逆にした画像を考えると...



白画素・黒画素の個数を数えると...(前のスライドの画像の白と黒を逆にした画像なので...)

黒画素10個

白画素4個

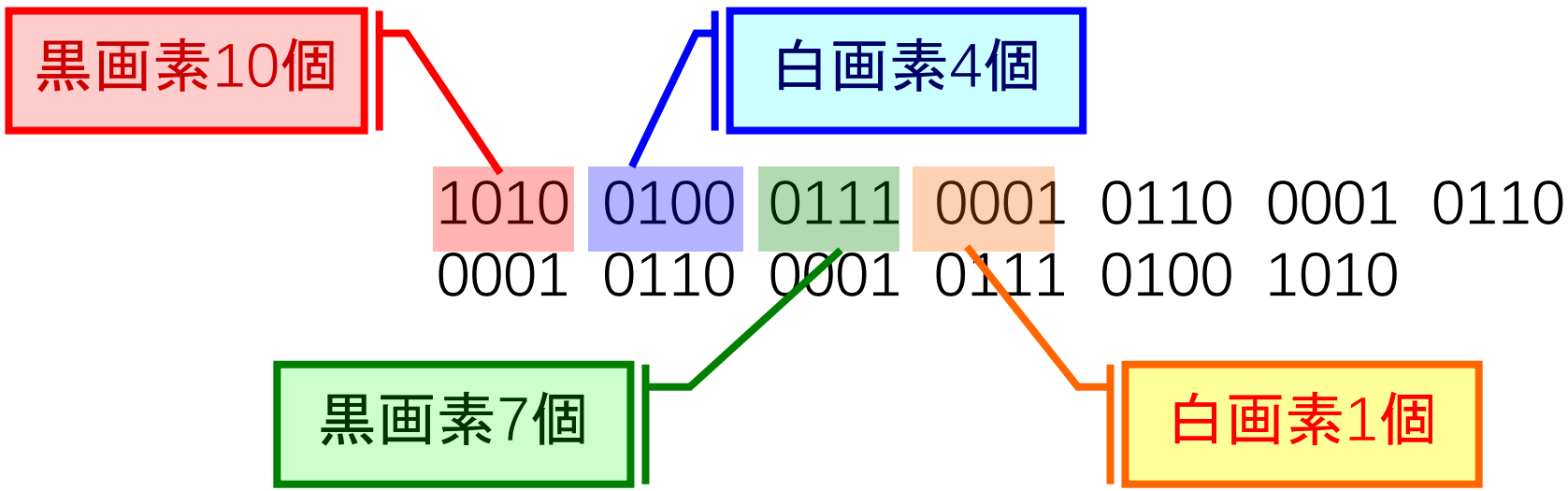
1010 0100 0111 0001 0110 0001 0110
0001 0110 0001 0111 0100 1010

黒画素7個

白画素1個

先頭が黒画素のとき[2]

- ランレングス符号化の考え方
  - 白画素・黒画素の個数だけを並べたとき、先頭の個数は白画素の個数とみなす



先頭の個数は黒画素の個数! …でもこれでは困る
→先頭の個数を「白画素0個」にすると良い

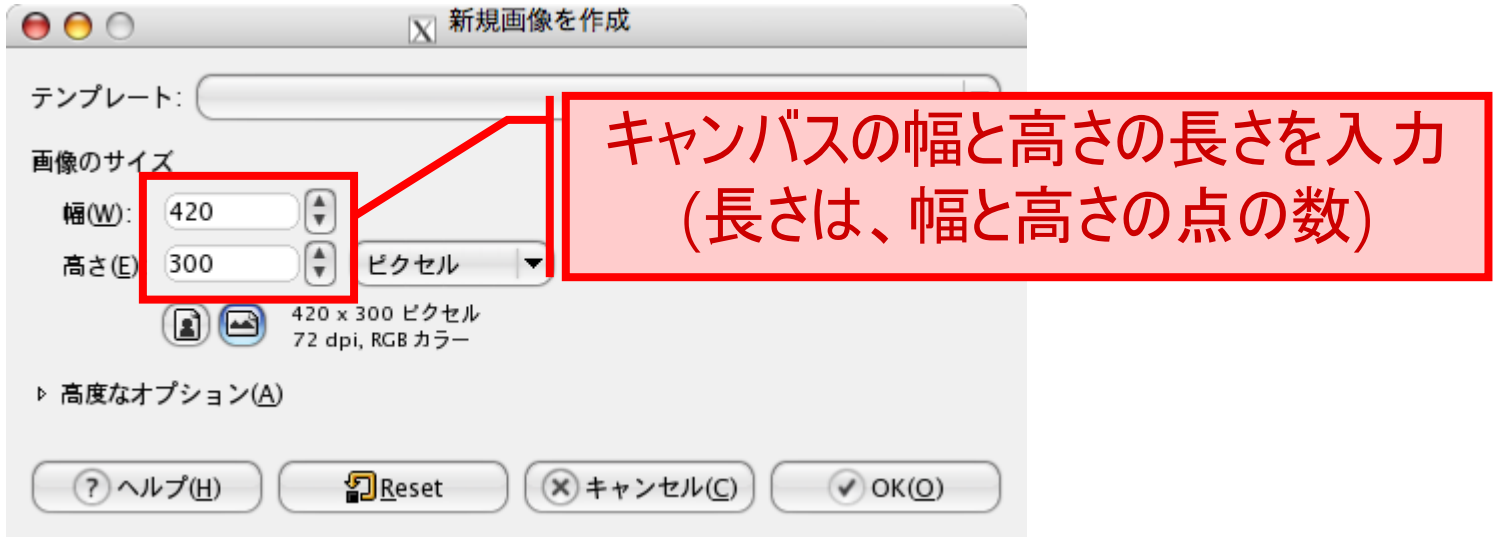
0000 0100 0111 0001 0110 0001 0110
0001 0110 0001 0111 0100 1010

Question!

圧縮の実習～準備: GIMP～

GIMPの基本～起動～

- GIMP: 描画ソフトウェア
- Finder→「アプリケーション」→「GIMP」をダブルクリック
- 「ファイル」→「新しい画像」をクリック
- 絵を描くキャンバスの大きさを決め、「OK」をクリック



GIMPの基本～描画～

- ツールボックスの中のボタンをクリックし、キャンバスに絵を描く
  - 鉛筆やブラシ、インクツールであれば、マウスをドラッグ＆ドロップ
    - ツールを選択すると、ツールボックスの下に線の太さなどの設定ができるウィンドウが表示
  - 文字であれば、キャンバス上でクリックし、文字を入力

鉛筆・ブラシ・文字のツール

クリックして線や文字の色を選択

GIMPの基本～描いたものの保存～

- キャンバスのメニュー→「ファイル」→「エクスポート」をクリック
  - 「ファイル」→「保存」ではないことに注意!
    - GIMPならではの形式で保存されてしまって、他のソフトウェアで開けなくなる
- 「名前」欄にファイル名を入力
  - ファイル名は、拡張子をつけて入力すること(どの拡張子で保存するかは後で説明)
    - GIMPは、ファイルの拡張子に応じてファイルの保存形式を決定
    - ファイル名を「image」としたいと思い、拡張子を「tiff」と指定された場合:「image.tiff」と入力する
- 「他のフォルダを参照」を押すと、保存するフォルダを選択可能

GIMPの基本～保存してある画像を開く～

- Finder→「アプリケーション」→「GIMP」をダブルクリック
- 「ファイル」→「開く/インポート」をクリック
- 表示されたウィンドウで、開きたいファイルを選択し、「開く」をクリック

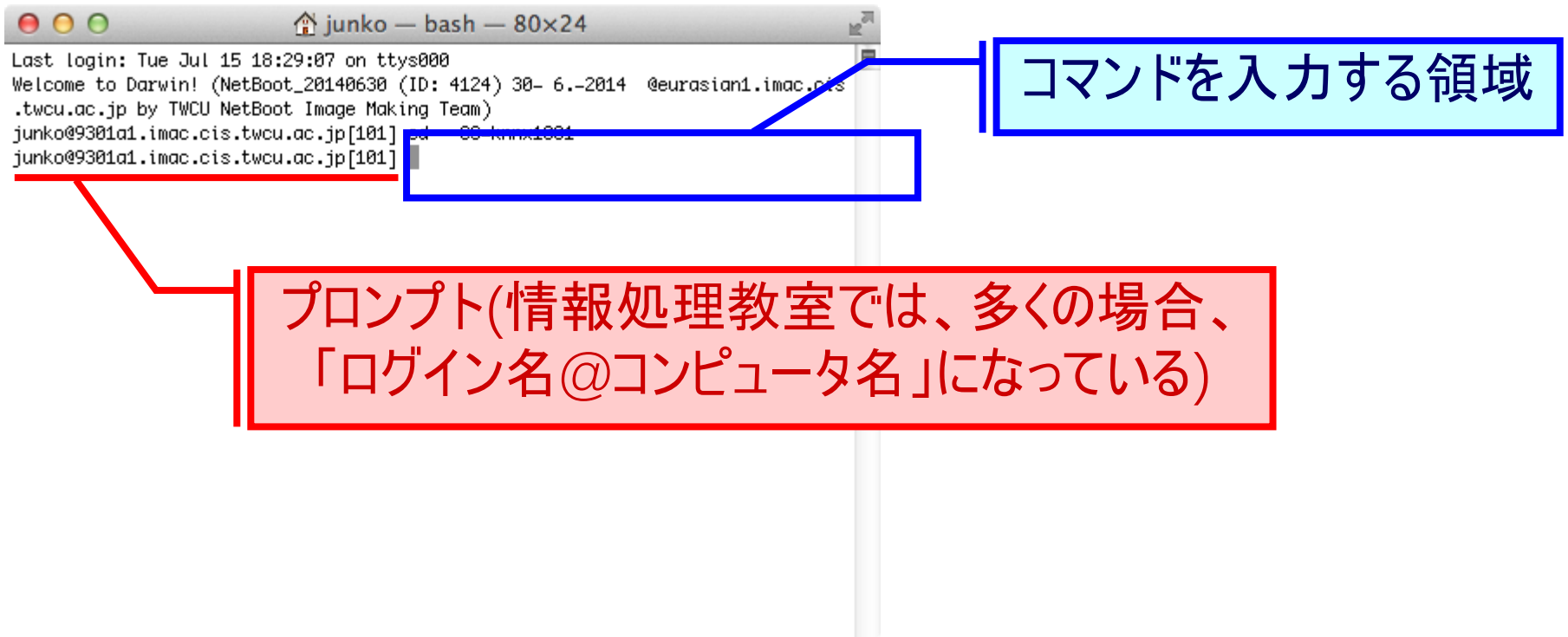
圧縮の実習～準備: ファイルサイズの確認方法～

ファイルサイズの確認方法

- Mac OS Xではリソースフォークがあるため、Finderからはファイルそのもののサイズを確認できない
  - リソースフォーク: アイコンの形や開くアプリケーションなど、ファイルに関する様々な情報を保存したファイル
    - Finderでは表示されないファイル
  - Finderで表示されるファイルサイズは、ファイルそのものサイズとリソースフォークのサイズをあわせたもの
    - Finderで表示されるファイルサイズは、圧縮の実習でのファイルサイズの比較には向かない

ファイルそのもののサイズを確認するには?(1)

- Finder→「ユーティリティ」→「ターミナル」を起動
  - 起動したウィンドウに、文字を入力して様々な処理をするためのソフトウェア
    - 入力する文字(命令)を「コマンド」と呼ぶ
    - コマンドは「プロンプト」の後に入力し、「Return」キーを押すことで実行される



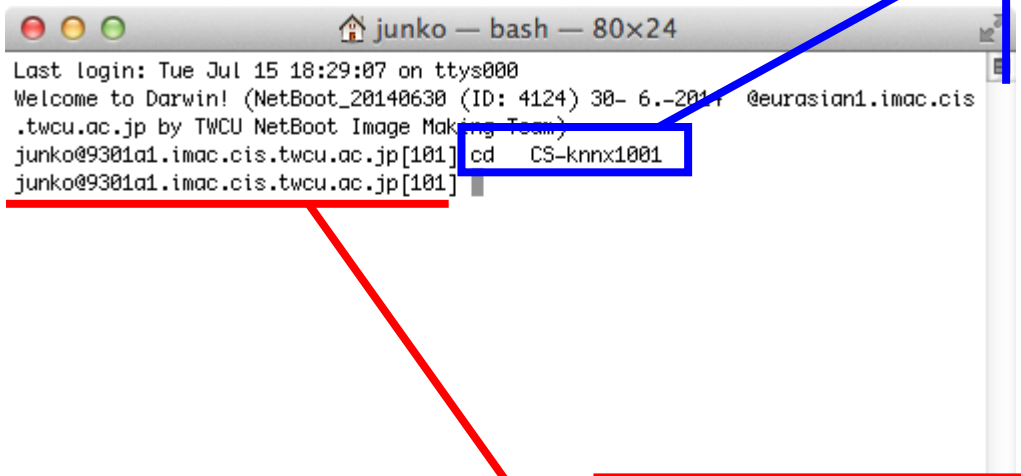
ファイルそのもののサイズを確認するには?(2)

- ターミナルでの作業場所を、ファイルを保存しているフォルダにあわせる
  - ターミナルでの作業場所は、前回ターミナルを使い終わった場所に設定されている
    - 初めて使った場合はホームフォルダ
  - ターミナルを起動したときに1度だけ行えば良い
- 作業場所を設定するには…

  cd　フォルダ名

  と、コマンド入力領域に入力し、「Return」キーを押す
  - フォルダ名は、「書類」は「Documents」、デスクトップは「Desktop」と入力
  - 「No such file or directory」など、何かのメッセージが表示されたら、やり直し
    - 以前に入力された内容は消せないので、そのまま追加で「cd　フォルダ名」を入力
  - 単に「cd」と入力して「Return」キーを押すと作業場所はホームフォルダに設定
    - どうしてもうまくいかない場合は、一度ホームフォルダに設定して再度やりなおし

ファイルそのもののサイズを確認するには?(3)



「cd CS-knnx1001」と入力し、作業場所を「CS-knnx1001」フォルダにあわせている

コマンドが成功したら、何もメッセージはなく次のプロンプトが表示される

- コマンドの実行に失敗したら、何かメッセージが表示されて次のプロンプトが表示される
- 失敗した場合は、前の内容は消せないので、新しくコマンドを書き直してやり直す

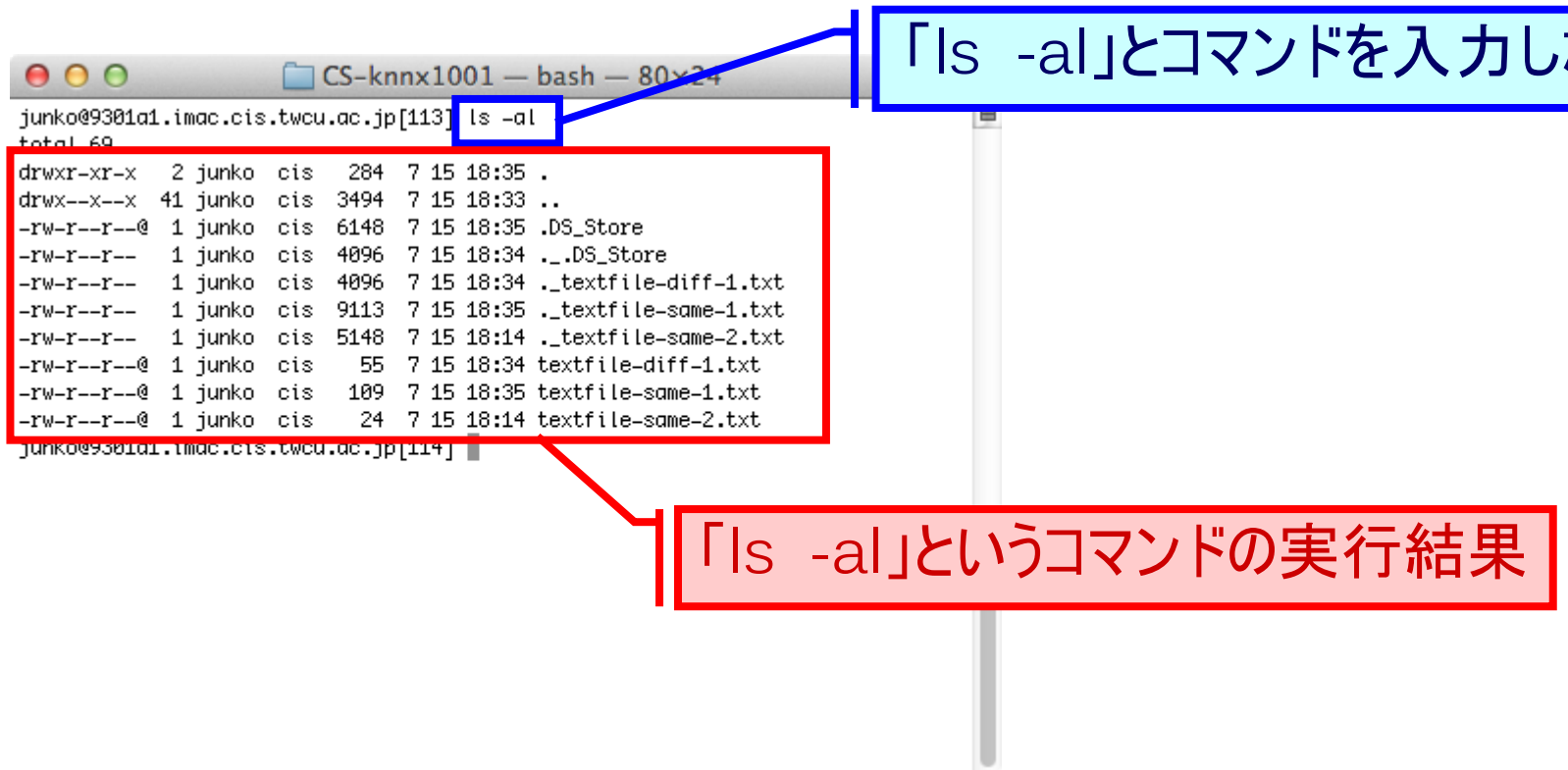
ファイルそのもののサイズを確認するには?(4)

- ファイルサイズを確認するには…

ls -al

とコマンド入力領域に入力し、「Return」キーを押す

  - 作業場所に設定したフォルダに保存されているすべてのファイルの情報が表示される



ファイルそのもののサイズを確認するには?(5)

- 「ls -al」コマンドで表示されたファイルサイズを確認
  - 「.」で始まるファイル名はリソースフォークなので関係なし

次回

- 実習をするので24102教室で授業
  - 圧縮の実習