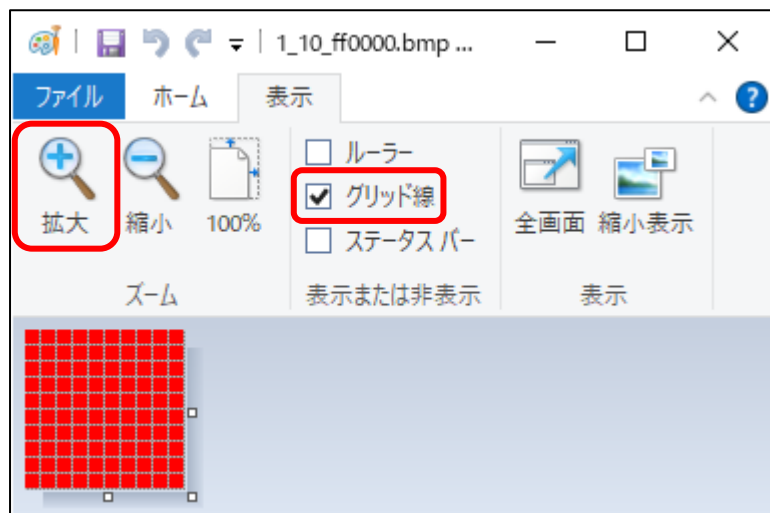


デジタル表現④ 画像のデジタル表現1 ベクタ形式(BMP ファイル)

・配布したフォルダの中のbmpフォルダの1つ目の10×10ドットの赤い画像を右クリックして、[プログラムから開く]→[ペイント]で、ペイントから表示させる

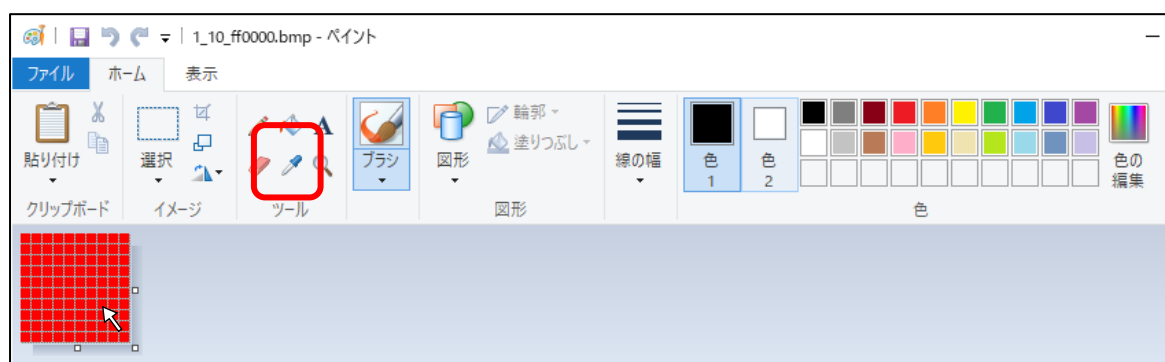


・Ctrl キーを押しながらマウスホイールを上回転がすか、表示タブの拡大ボタンを押して、最大倍率にして、グリッド線にチェックをして、いくつかの点で構成されているが確認する。



この画像ファイルは縦横10ドットあるので、100個の点で構成されている。

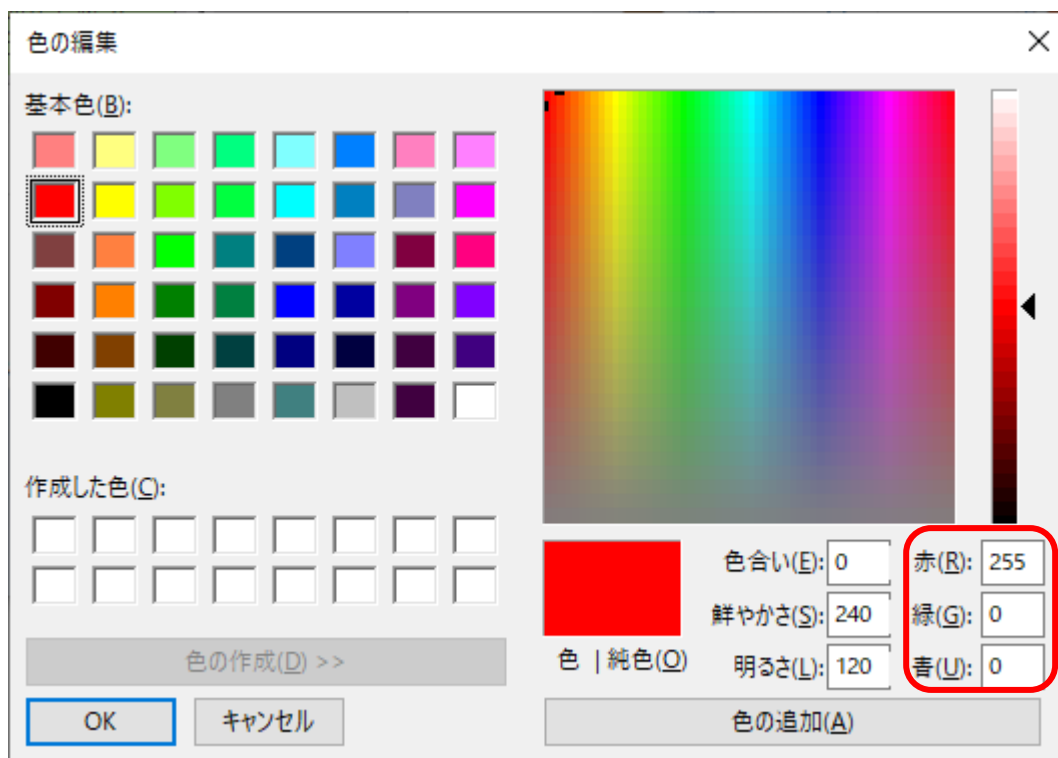
・ホームタブのスポイトツールを選択し、画像をクリックする。



・色1に吸い取った色が設定されていることを確認し、「色の編集」ボタンをクリックする。



・色1の赤と緑と青の大きさが10進数で表示される。16進数ではどのように表示されるか考える。



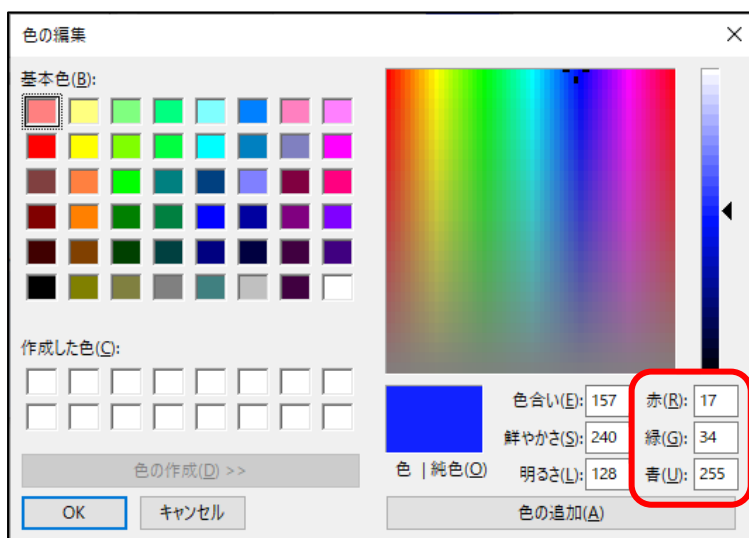
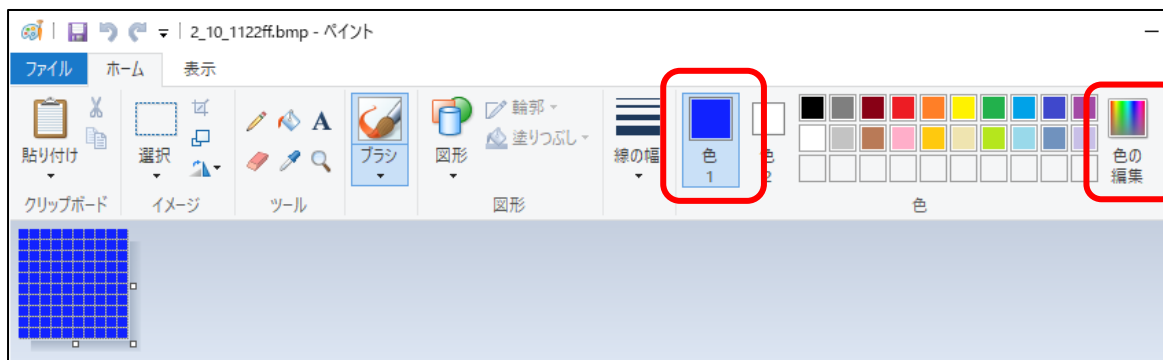
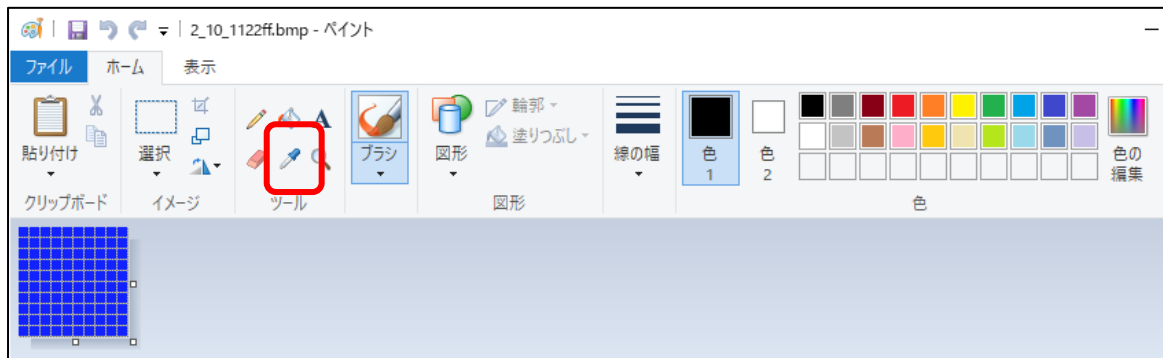
・BzEditor を起動して、1 つ目のファイルをウィンドウにドラッグして開くと、FF 00 00 のパターンが繰り返し出てくることが分かる。このうちの FF が赤、00 が緑と青であることが推測できる。

[illegible]

- ・RGB の順番が分からないので、2つ目のファイルを使う。



- ・先ほどと同じ要領でペイントで開き、RGB の値を確認する



赤:17 緑:34 青:255

となっており、これらの値は16進数で表現するとどのようなになるか考える。

・画像ファイルを BzEditor で表示させる

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0	42	4D	76	01	00	00	00	00	00	00	36	00	00	00	28	00
10	00	00	0A	00	00	00	0A	00	00	00	01	00	18	00	00	00
20	00	00	40	01	00	00	00	00	00	00	00	00	00	00	00	00
30	00	00	00	00	00	00	FF	22	11	FF	22	11	FF	22	11	FF
40	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
50	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
60	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
70	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
80	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
90	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
A0	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
B0	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
C0	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
D0	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
E0	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
F0	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
100	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
110	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
120	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
130	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
140	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
150	11	FF	22	11	00	00	FF	22	11	FF	22	11	FF	22	11	FF
160	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22	11	FF	22
170	11	FF	22	11	00	00										

・FF 22 11 のパターンが繰り返し表示されており、青、緑、赤の順に記録されていることがわかる。

000070	00	00	00	00	00	00	00	00-00	00	11	11	11	22	22	22
000080	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
000090	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
0000A0	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
0000B0	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
0000C0	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
0000D0	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
0000E0	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
0000F0	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
000100	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
000110	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
000120	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
000130	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
000140	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
000150	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
000160	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
000170	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
000180	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
000190	88	88	99	99	99	AA	AA	AA-00	00	11	11	11	22	22	22
0001A0	33	33	33	44	44	44	55	55-55	66	66	66	77	77	77	88
0001B0	88	88	99	99	99	AA	AA	AA-00	00						

000070	00	00	00	00	00	00	00	00	00	11	11	11	22	22	22
000080	32	22	22	44	44	44	55	55	55	66	66	66	77	77	77
000090	88	88	88	88	88	AA	AA	AA	00	00	11	11	11	22	22
0000A0	38	88	88	44	44	44	55	55	55	88	88	88	77	77	77
0000B0	88	88	88	88	88	AA	AA	AA	00	00	11	11	11	22	22
0000C0	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
0000D0	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
0000E0	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
0000F0	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
000100	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
000110	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
000120	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
000130	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
000140	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
000150	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
000160	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
000170	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
000180	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
000190	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22
0001A0	33	33	33	44	44	44	55	55	55	66	66	66	77	77	77
0001B0	88	88	99	99	99	AA	AA	AA	00	00	11	11	11	22	22

問 このファイルのデータ量は何バイトか求めよ？

解答

1ドットがRGBそれぞれ1バイトで記録しているので、
1ドットのデータ量は3バイト。それが10×10あるのだから、

$3 \times 10 \times 10 = 300$ バイト
となる。

(考察)

ファイルを右クリックしてプロパティを選択してファイルサイズを見ると、374バイトになっている。RGB の
大きさのデータの他に、この画像ファイルを構成する何らかの情報が記録されていることが分かる。
少なくとも、横の点の数が分からないと平面の画像にならないので、横方向のドット数の情報は必要である。



SVG 画像(ベクタ形式)について

SVG 画像の1つ目のファイルを右クリックしてプログラムから開くから、メモ帳で表示させると



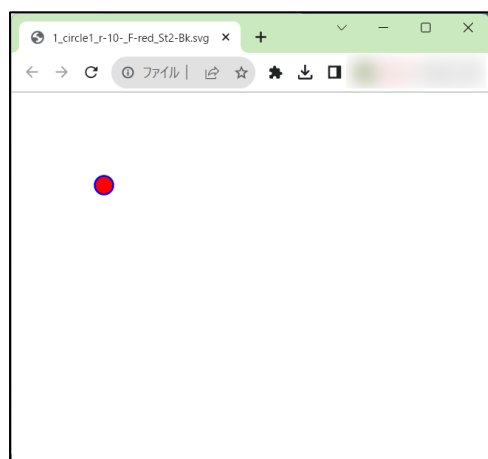
```
<svg
  xmlns="http://www.w3.org/2000/svg"
  version="1.1"
  width="400" height="400">

  <circle style="fill:#ff0000;stroke:#0000ff;stroke-width:2"
    cx="100"
    cy="100" r="10" />

</svg>
```

と表示される。

再び、ファイルを右クリックして、ファイルから開くから「Google Chrome」を選択すると、

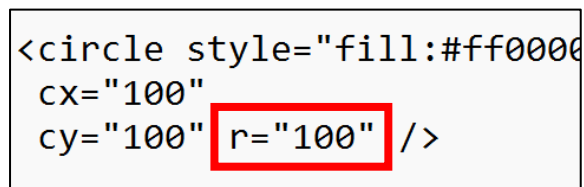


と表示される

circle タグの、fill は塗りつぶしの色、stroke は線の色、stroke-width は線の太さ、cxは中心のx座標、cyは中心のy座標、rは半径を意味している。

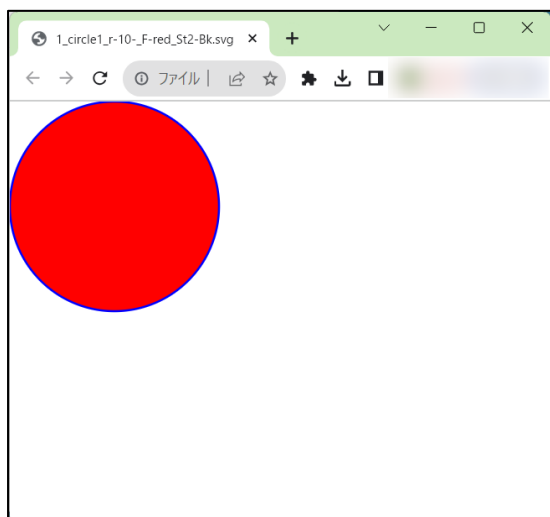
円を大きくするときはrの値を変える。

メモ帳で



```
<circle style="fill:#ff0000
  cx="100"
  cy="100" r="100" />
```

のようにrの値を100にして上書きして保存し、Google Chrome で再読み込みすると



と表示される。

円が大きくなっているが、ファイルサイズは文字の「10」が「100」になっただけなので、1文字分のデータ量しか増えていない。ラスタ形式の場合はその文、点を増やさないとならないので、ファイルサイズが大きくなる。また、小さいものを大きく拡大した場合、ラスタ形式の場合が点が大きくなるので、ギザギザが目立つようになるが、ベクタ形式の場合は再描画されるので、そのようなことにはならない。