［演習 06］画像情報処理（ウォーミングアップ）

学生番号 ： 20216187
氏名 ： 劉潤之
提出日 ： 2022/11/03

［レポート作成の準備］

1. script コマンドを用いて，プログラムリスト・コンパイル結果・実行結果を一つのファイルに書き出しておく．（演習問題ごとに，ファイルを書き出しておくこと．）
2. レポートに画像を載せる必要がある場合には，画像を準備しておく．（どのような図を記載すべきかについては問題文に示されている．）

［レポート作成方法］

次ページ以降において，演習問題ごとに，以下を実施すること．

1. 問題番号を，ドロップダウンリストより 2 箇所選択する．
2. プログラムリスト・コンパイル結果・実行結果を，準備しておいたファイルからコピーし，所定の場所に，テキスト形式でペーストする．
3. 必要に応じて，所定の場所に図を挿入するとともに，その題目や説明を記入する．

なお，未使用なページやスペースは，そのままにしておいてよい（削除しなくてよい）．

［レポート提出方法］

すべての演習問題を終了した後，レポートを作成して提出する．ここで，レポートの提出期限は，原則として，次回演習日の前夜までとする．

提出は，工学部の「ポータルサイト」の課題提出のページから実施すること．なお，提出の際，コメントを特に記載する必要はない．

［提出ファイル］

提出は，PDF ファイルとする．（word でレポートを作成し，一旦保存する．その後，同ファイルを「名前を付けて保存（コピーを保存)」する．ここで，“ファイル名”の下にある“ファイルの種類”を PDF とすれば，PDF ファイルが作成できる．）

なお，ファイル名は，“01_xxxxxxxx.pdf”とする．ここで，先頭の 2 桁の値は演習問題の回，末尾の xxxxxxxx の部分は学生番号とする．

【問題 6-A-1】

［プログラムリスト・コンパイル結果・実行結果］

```
Script started on Thu Nov  3 14:25:24 2022
u20216187@gw[31]: cat 06_A_01.c

#include<stdio.h>
#include<stdlib.h>
#include<math.h>

#define MAX 100
#define CH 3
#define Ych 0
#define ROW 3
#define COL 3

unsigned char header[54];
unsigned char imgin[3][512][512];//入力画像
unsigned char imgout[3][512][512];//
double dtemp[3][512][512];
int itemp[3][512][512];

void processing();
void get_data();
void put_data();
void rgb_to_ybr();
void ybr_to_rgb();

int width, height;//画像の幅と高さ
int bits;//画像のビット数

double rgb_con_ybr[ROW][COL] = {
  { 0.2990, 0.5870, 0.1140},
   {-0.1687,-0.3313, 0.5000},
   { 0.5000,-0.4187,-0.0813}
};
double ybr_con_rgb[ROW][COL] = {
  { 1.0000, 0.0000, 1.4020},
  { 1.0000,-0.3441,-0.7141},
  { 1.0000, 1.7720, 0.0000}
};
int main() {
    get_data();
    rgb_to_ybr();
    processing();
    ybr_to_rgb();
    put_data();
    return 0;
}

int calculate(int a, int b);//バイト数計算
void get_data() {
    FILE* fp;
```

```
  char filename[MAX];
  int c;
  int filesize, offset, bite_px;//画像の属性


  printf("ファイル名を入力して下さい:");
  scanf("%s", filename);
  fp = fopen(filename, "rb");
  if (fp == NULL) {
      printf("%s をオーブンできません¥n", filename);
      exit(1);
  }
  printf("%s をオーブンしました.¥n", filename);
  for (int i = 0; i < 54; i++) {
      c = fgetc(fp);
      header[i] = c;
  }
  /* printf("¥n¥n<ファイルタイプ>¥n");
  for (int i = 0; i < 2; i++) {
    printf("header[%d]=%02x ", i, header[i]);
    } */
  printf("¥n¥n ファイルサイズ>¥n");
  filesize = calculate(5, 2);
/*  for (int i = 2; i < 6; i++) {
      printf("header[%d]=%02x ", i, header[i]);
  }*/
  printf("¥n¥n%d バイト", filesize);
  /* printf("¥n¥n<予約領域>¥n");
  for (int i = 6; i < 10; i++) {
    printf("header[%d]=%02x ", i, header[i]);
    } */
  printf("¥n¥n<オフセット>¥n");
  /*for (int i = 10; i < 14; i++) {
      printf("header[%d]=%02x ", i, header[i]);

  }*/
  offset = calculate(13, 10);
  printf("¥n¥n%d バイト", offset);
  /*  printf("¥n¥n<情報ヘッダサイズ>¥n");
  for (int i = 14; i < 18; i++) {
    printf("header[%d]=%02x ", i, header[i]);
    } */
  printf("¥n¥n<画像の幅>¥n");
  width = calculate(21, 18);
  /*for (int i = 18; i < 22; i++) {
      printf("header[%d]=%02x ", i, header[i]);

  }*/
  printf("¥n%d 画素¥n", width);
  printf("¥n¥n<画像の高さ>");
  height = calculate(25, 22);
  /*for (int i = 22; i < 26; i++) {
      printf("header[%d]=%02x ", i, header[i]);
  }*/

  printf("¥n%d ライン¥n", height);
  /*  printf("¥n¥n<色プレーン数>¥n");
```

```
    for (int i = 26; i < 28; i++) {
      printf("header[%d]=%02x ", i, header[i]);
      } */
    printf("¥n¥n<1 画素にあたりのビット数>¥n");
    bite_px = calculate(29, 28);
  /*  for (int i = 28; i < 30; i++) {
        printf("header[%d]=%02x ", i, header[i]);

    }*/
    printf("¥n%d ビット", bite_px);
    /* printf("¥n¥n<圧縮方式>¥n");
    for (int i = 30; i < 34; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<画像データサイズ>¥n");
    for (int i = 34; i < 38; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<水平解像度>¥n");
    for (int i = 39; i < 42; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<垂直解像度>¥n");
    for (int i = 42; i < 46; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<色数>¥n");
    for (int i = 46; i < 50; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<重要な色数>¥n");
    for (int i = 50; i < 54; i++) {
      printf("header[%d]=%02x ", i, header[i]);
      } */
    printf("¥n¥n<挿入ビット数>¥n");
    bits = (offset + width * height * (bite_px / 8)) % 4;
    printf("%d バイト¥n", bits);
    //imgin[][][]の初期化
    for (int h = height - 1; h >= 0; h--) {
        for (int w = 0; w < width; w++) {
            for (int i = 2; i >= 0; i--) {
                imgin[i][w][h] = (unsigned char)fgetc(fp);
            }
        }
    }

    fclose(fp);
    printf("¥n%s をクローズしました.¥n", filename);

}
int calculate(int a, int b) {
    int value;
    value = header[a];
    for (int i = a - 1; i >= b; i--) {
        value <<= 8;
        value += header[i];
    }
```

```
    return value;

}

void processing() {

    /* for (int y=height-1;y>=0;y--){
   for(int x=0;x<width;x++){
     for(int i=3;i>0;i--){
       imgout[i][x][y]=imgin[i][width-1-x][y];
     }
   }
 }*/


    printf("出力画像データを作成しました.¥n");
}
void put_data() {
    FILE* fp;
    char cpfilename[MAX];//出力画像名前

    printf("出力ファイル名を入力してください:");
    scanf(" %s", cpfilename);
    fp = fopen(cpfilename, "wp");
    printf("%s をオープンしました.¥n", cpfilename);
    for (int i = 0; i < 54; i++) {
        fputc(header[i], fp);

    }

    for (int h = height - 1; h >= 0; h--) {
        for (int w =width; w > 0; w--) {
            for (int i = 2; i >= 0; i--) {
                fputc(imgout[i][w][h], fp);
            }
        }
    }
    for (int i = 0; i < bits; i++) {
        fputc('¥0', fp);
    }
    fclose(fp);
    printf("%s をクローズしました.¥n", cpfilename);
}

void rgb_to_ybr() {
    int i, x, y;
    /*if (height <= 16 || width <= 16) {

        printf("¥n<R 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<G 信号>¥n");
```

```
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            printf("%02x ", imgin[1][x][y]);
        }
        printf("¥n");
    }

    printf("¥n<B 信号>¥n");
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            printf("%02x ", imgin[2][x][y]);
        }
        printf("¥n");
    }
}
else {
    printf("画像サイズが大きいため表示しません.¥n");
}*/

for (y = 0; y < height; y++) {
    for (x = 0; x < width; x++) {

        for (i = 0; i < 3; i++) {
            dtemp[i][x][y] = 0.0;
            for (int j = 0; j < 3; j++)
                dtemp[i][x][y] += rgb_con_ybr[i][j] * (double)imgin[j][x][y];
        }
    }
}

for (y = 0; y < height; y++) {
    for (x = 0; x < width; x++) {
        for (i = 0; i < 3; i++) {

            if (dtemp[i][x][y] > 0.0) {
                itemp[i][x][y] = (int)(dtemp[i][x][y] + 0.5);
            }
            else {
                itemp[i][x][y] = (int)(dtemp[i][x][y] - 0.5);
            }

            if (i != 0) {
                itemp[i][x][y] += 128;
            }

            if (itemp[i][x][y] > 255) {
                itemp[i][x][y] = 255;
            }
            else if (itemp[i][x][y] < 0) {
                itemp[i][x][y] = 0;
            }

            imgin[i][x][y] = (unsigned char)itemp[i][x][y];
        }
    }
}
```

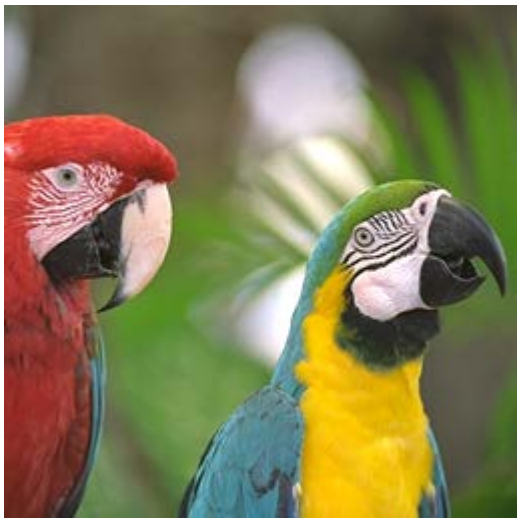
```
    /*printf("¥n<入力 YCbCr 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<Y 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cb 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cr 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

}

void ybr_to_rgb() {
    int i, x, y, j;

    printf("¥n<入力 YCbCr 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<Y 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cb 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[1][x][y]);
            }
            printf("¥n");
        }
```

```
        printf("¥n<Cr 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }

    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {
                dtemp[i][x][y] = 0.0;
                for (j = 0; j < 3; j++)
                    if (j == 0)
                        dtemp[i][x][y]        +=        ybr_con_rgb[i][j]        *
(double)imgin[j][x][y];
                    else
                        dtemp[i][x][y]        +=        ybr_con_rgb[i][j]        *
(double)(imgin[j][x][y] - 128);
            }
        }
    }


    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {

                if (dtemp[i][x][y] > 0.0) {
                    itemp[i][x][y] = (int)(dtemp[i][x][y] + 0.5);
                }
                else {
                    itemp[i][x][y] = (int)(dtemp[i][x][y] - 0.5);
                }

                if (itemp[i][x][y] > 255) {
                    itemp[i][x][y] = 255;
                }
                else if (itemp[i][x][y] < 0) {
                    itemp[i][x][y] = 0;
                }

                imgout[i][x][y] = (unsigned char)itemp[i][x][y];
            }
        }
    }

    /*printf("¥n<入力 RGB 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<R 信号>¥n");
        for (y = 0; y < height; y++) {
```

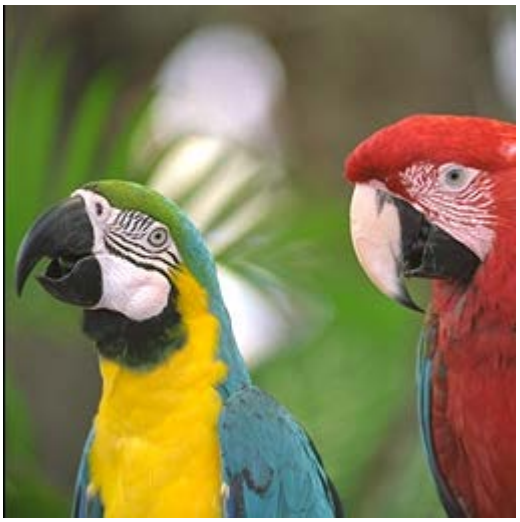
```
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<G 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<B 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

}
u20216187@gw[32]: gcc -Wall 06_A_01.c  -o 06_ A_01.c [K[K[K

u20216187@gw[33]: ./06_A_01

ファイル名を入力して下さい:parrots.bmp
parrots.bmp をオープンしました.


ファイルサイズ>


196664 バイト

<オフセット>


54 バイト

<画像の幅>

256 画素


<画像の高さ>
256 ライン


<1 画素にあたりのビット数>

24 ビット
```

```
<挿入ビット数>
2 バイト

parrots.bmp をクローズしました.
出力画像データを作成しました.

<入力 YCbCr 信号(整数値)>
画像サイズが大きいため表示しません.
出力ファイル名を入力してください:parrots_06_A_1.bmp
parrots_06_A_1.bmp をオープンしました.
parrots_06_A_1.bmp をクローズしました.
u20216187@gw[34]: exit

exit

Script done on Thu Nov  3 14:26:42 2022
```

［添付図］



Parrots.bmp



Parrots_06_A_1.bmp

【問題 6-B-1】

［プログラムリスト・コンパイル結果・実行結果］

```
Script started on Thu Nov  3 14:26:55 2022
u20216187@gw[31]: cat 6-B-1.c

#include<stdio.h>
#include<stdlib.h>
#include<math.h>

#define MAX 100
#define CH 3
#define Ych 0
#define ROW 3
#define COL 3

unsigned char header[54];
unsigned char imgin[3][512][512];//入力画像
unsigned char imgout[3][512][512];//
double dtemp[3][512][512];
int itemp[3][512][512];

void processing();
void get_data();
void put_data();
void rgb_to_ybr();
void ybr_to_rgb();

int width, height;//画像の幅と高さ
int bits;//画像のビット数

double rgb_con_ybr[ROW][COL] = {
  { 0.2990, 0.5870, 0.1140},
   {-0.1687,-0.3313, 0.5000},
   { 0.5000,-0.4187,-0.0813}
};
double ybr_con_rgb[ROW][COL] = {
  { 1.0000, 0.0000, 1.4020},
  { 1.0000,-0.3441,-0.7141},
  { 1.0000, 1.7720, 0.0000}
};
char ybr_name[3][3] = { "Y","Cb","Cr" };

int main() {
    get_data();
    rgb_to_ybr();
    processing();
    ybr_to_rgb();
    put_data();
    return 0;
}

int calculate(int a, int b);//バイト数計算
void get_data() {
```

```
FILE* fp;

char filename[MAX];
int c;
int filesize, offset, bite_px;//画像の属性


printf("ファイル名を入力して下さい:");
scanf("%s", filename);
fp = fopen(filename, "rb");
if (fp == NULL) {
    printf("%s をオープンできません¥n", filename);
    exit(1);
}
printf("%s をオープンしました.¥n", filename);
for (int i = 0; i < 54; i++) {
    c = fgetc(fp);
    header[i] = c;
}
/* printf("¥n¥n<ファイルタイプ>¥n");
for (int i = 0; i < 2; i++) {
  printf("header[%d]=%02x ", i, header[i]);
  } */
printf("¥n¥n ファイルサイズ>¥n");
filesize = calculate(5, 2);
/*  for (int i = 2; i < 6; i++) {
      printf("header[%d]=%02x ", i, header[i]);
  }*/
printf("¥n¥n%d バイト", filesize);
/* printf("¥n¥n<予約領域>¥n");
for (int i = 6; i < 10; i++) {
  printf("header[%d]=%02x ", i, header[i]);
  } */
printf("¥n¥n<オフセット>¥n");
/*for (int i = 10; i < 14; i++) {
    printf("header[%d]=%02x ", i, header[i]);

}*/
offset = calculate(13, 10);
printf("¥n¥n%d バイト", offset);
/*  printf("¥n¥n<情報ヘッダサイズ>¥n");
for (int i = 14; i < 18; i++) {
  printf("header[%d]=%02x ", i, header[i]);
  } */
printf("¥n¥n<画像の幅>¥n");
width = calculate(21, 18);
/*for (int i = 18; i < 22; i++) {
    printf("header[%d]=%02x ", i, header[i]);

}*/
printf("¥n%d 画素¥n", width);
printf("¥n¥n<画像の高さ>");
height = calculate(25, 22);
/*for (int i = 22; i < 26; i++) {
    printf("header[%d]=%02x ", i, header[i]);
}*/
```

```
    printf("¥n%d ライン¥n", height);
    /*  printf("¥n¥n<色プレーン数>¥n");
    for (int i = 26; i < 28; i++) {
      printf("header[%d]=%02x ", i, header[i]);
      } */
    printf("¥n¥n<1 画素にあたりのビット数>¥n");
    bite_px = calculate(29, 28);
    /*  for (int i = 28; i < 30; i++) {
          printf("header[%d]=%02x ", i, header[i]);

      }*/
    printf("¥n%d ビット", bite_px);
    /* printf("¥n¥n<圧縮方式>¥n");
    for (int i = 30; i < 34; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<画像データサイズ>¥n");
    for (int i = 34; i < 38; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<水平解像度>¥n");
    for (int i = 39; i < 42; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<垂直解像度>¥n");
    for (int i = 42; i < 46; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<色数>¥n");
    for (int i = 46; i < 50; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<重要な色数>¥n");
    for (int i = 50; i < 54; i++) {
      printf("header[%d]=%02x ", i, header[i]);
      } */
    printf("¥n¥n<挿入ビット数>¥n");
    bits = (offset + width * height * (bite_px / 8)) % 4;
    printf("%d バイト¥n", bits);
    //imgin[][][]の初期化
    for (int h = height - 1; h >= 0; h--) {
        for (int w = 0; w < width; w++) {
            for (int i = 2; i >= 0; i--) {
                imgin[i][w][h] = (unsigned char)fgetc(fp);
            }
        }
    }

    fclose(fp);
    printf("¥n%s をクローズしました.¥n", filename);

}
int calculate(int a, int b) {
    int value;
    value = header[a];
    for (int i = a - 1; i >= b; i--) {
        value <<= 8;
```

```
        value += header[i];
    }
    return value;

}

void processing() {

    /* for (int y=height-1;y>=0;y--){
   for(int x=0;x<width;x++){
     for(int i=3;i>0;i--){
       imgout[i][x][y]=imgin[i][width-1-x][y];
     }
   }
 }*/

    int i, x, y;
    int copy[3];

    printf("¥n コピーモードを入力して下さい.¥n");
    printf("(コピーする場合 : 1, 固定値に置き換える場合 : 0)¥n");
    for (i = 0; i < 3; i++) {
        printf("%-2s : ", ybr_name[i]);
        scanf("%d", &copy[i]);
    }

    for (i = 0; i < 3; i++) {
        if (copy[i] == 1) {
            for (y = 0; y < height; y++) {
                for (x = 0; x < width; x++) {
                    imgout[i][x][y] = imgin[i][x][y];
                }
            }
        }
        else {
            for (y = 0; y < height; y++) {
                for (x = 0; x < width; x++) {
                    imgout[i][x][y] = 128;
                }
            }
        }
    }


    printf("出力画像データを作成しました.¥n");
}
void put_data() {
    FILE* fp;
    char filename1[MAX];
    int i, x, y;

    printf("出力ファイル名を入力して下さい:");
    scanf("%s", filename1);

    fp = fopen(filename1, "wp");

    printf("%s をオープンしました.¥n", filename1);
```

```
    for (i = 0; i < 54; i++) {
        fputc(header[i], fp);
    }


    for (y = height - 1; y >= 0; y--) {
        for (x = 0; x < width; x++) {
            for (i = 2; i >= 0; i--) {
                fputc(imgout[i][x][y], fp);
            }
        }
    }

    for (i = 0; i < bits; i++) {
        fputc('¥0', fp);
    }

    fclose(fp);

    printf("%s をクローズしました.¥n", filename1);

}
void rgb_to_ybr() {
    int i, x, y;
    /*if (height <= 16 || width <= 16) {

        printf("¥n<R 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<G 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<B 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
```

```
        for (i = 0; i < 3; i++) {
            dtemp[i][x][y] = 0.0;
            for (int j = 0; j < 3; j++)
                dtemp[i][x][y] += rgb_con_ybr[i][j] * (double)imgin[j][x][y];
        }
    }
}

for (y = 0; y < height; y++) {
    for (x = 0; x < width; x++) {
        for (i = 0; i < 3; i++) {

            if (dtemp[i][x][y] > 0.0) {
                itemp[i][x][y] = (int)(dtemp[i][x][y] + 0.5);
            }
            else {
                itemp[i][x][y] = (int)(dtemp[i][x][y] - 0.5);
            }

            if (i != 0) {
                itemp[i][x][y] += 128;
            }

            if (itemp[i][x][y] > 255) {
                itemp[i][x][y] = 255;
            }
            else if (itemp[i][x][y] < 0) {
                itemp[i][x][y] = 0;
            }

            imgin[i][x][y] = (unsigned char)itemp[i][x][y];
        }
    }
}

/*printf("¥n<入力 YCbCr 信号(整数値)>¥n");

if (height <= 16 || width <= 16) {

    printf("¥n<Y 信号>¥n");
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            printf("%02x ", imgin[0][x][y]);
        }
        printf("¥n");
    }

    printf("¥n<Cb 信号>¥n");
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            printf("%02x ", imgin[1][x][y]);
        }
        printf("¥n");
    }

    printf("¥n<Cr 信号>¥n");
```

```
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

}

void ybr_to_rgb() {
    int i, x, y, j;

   /* printf("¥n<入力 YCbCr 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<Y 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cb 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cr 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {
                dtemp[i][x][y] = 0.0;
                for (j = 0; j < 3; j++)
                    if (j == 0)
                        dtemp[i][x][y]        +=        ybr_con_rgb[i][j]        *
(double)imgin[j][x][y];
                    else
```

```
                    dtemp[i][x][y]       +=       ybr_con_rgb[i][j]       *
(double)(imgin[j][x][y] - 128);
            }
        }
    }


    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {

                if (dtemp[i][x][y] > 0.0) {
                    itemp[i][x][y] = (int)(dtemp[i][x][y] + 0.5);
                }
                else {
                    itemp[i][x][y] = (int)(dtemp[i][x][y] - 0.5);
                }

                if (itemp[i][x][y] > 255) {
                    itemp[i][x][y] = 255;
                }
                else if (itemp[i][x][y] < 0) {
                    itemp[i][x][y] = 0;
                }

                imgout[i][x][y] = (unsigned char)itemp[i][x][y];
            }
        }
    }

    /*printf("¥n<入力 RGB 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<R 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<G 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<B 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[2][x][y]);
            }
            printf("¥n");
        }
```

```
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

}
u20216187@gw[32]: gcc 6-B-1.c -o 6-B-1

u20216187@gw[33]: ./6-B-1

ファイル名を入力して下さい:pepper.bmp
pepper.bmp をオーブンしました.


ファイルサイズ>


196664 バイト

<オフセット>


54 バイト

<画像の幅>

256 画素


<画像の高さ>
256 ライン


<1 画素にあたりのビット数>

24 ビット

<挿入ビット数>
2 バイト

pepper.bmp をクローズしました.

 コピーモードを入力して下さい.
(コピーする場合 : 1, 固定値に置き換える場合 : 0)
Y  : 1
Cb : 0
Cr : 0
出力画像データを作成しました.
出力ファイル名を入力して下さい:pepper_06_B_1_1.bmp
pepper_06_B_1_1.bmp をオープンしました.
pepper_06_B_1_1.bmp をクローズしました.
u20216187@gw[34]: ./6-B-1

ファイル名を入力して下さい:pepper.bmp
pepper.bmp をオーブンしました.
```

ファイルサイズ>

196664 バイト

<オフセット>

54 バイト

<画像の幅>

256 画素

<画像の高さ>
256 ライン

<1 画素にあたりのビット数>

24 ビット

<挿入ビット数>
2 バイト

pepper.bmp をクローズしました.

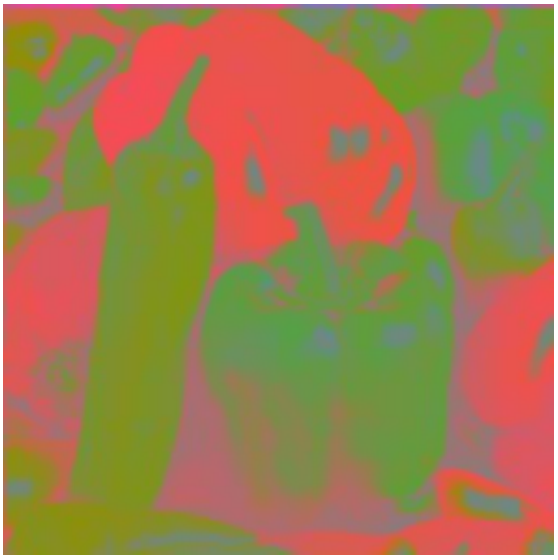
コピーモードを入力して下さい.
(コピーする場合 : 1, 固定値に置き換える場合 : 0)
Y : 0
Cb : 1
Cr : 1
出力画像データを作成しました.
出力ファイル名を入力して下さい:pepper_06_B_1_2.bmp
pepper_06_B_1_2.bmp をオープンしました.
pepper_06_B_1_2.bmp をクローズしました.
u20216187@gw[35]: exit

exit

Script done on Thu Nov 3 14:28:39 2022

［添付図］

Pepper_06_B_1_1.bmp



Pepper_06_B_1_2.bmp

【問題 6-B-2】

［プログラムリスト・コンパイル結果・実行結果］

```
Script started on Thu Nov  3 14:28:50 2022
u20216187@gw[31]: cat 6-B-2.c

#include<stdio.h>
#include<stdlib.h>
#include<math.h>

#define MAX 100
#define CH 3
#define Ych 0
#define ROW 3
#define COL 3

unsigned char header[54];
unsigned char imgin[3][512][512];//入力画像
unsigned char imgout[3][512][512];//
double dtemp[3][512][512];
int itemp[3][512][512];

void processing();
void get_data();
void put_data();
void rgb_to_ybr();
void ybr_to_rgb();

int width, height;//画像の幅と高さ
int bits;//画像のビット数

double rgb_con_ybr[ROW][COL] = {
  { 0.2990, 0.5870, 0.1140},
   {-0.1687,-0.3313, 0.5000},
   { 0.5000,-0.4187,-0.0813}
};
double ybr_con_rgb[ROW][COL] = {
  { 1.0000, 0.0000, 1.4020},
  { 1.0000,-0.3441,-0.7141},
  { 1.0000, 1.7720, 0.0000}
};
//char ybr_name[3][3] = { "Y","Cb","Cr" };

int main() {
    get_data();
    rgb_to_ybr();
    processing();
    ybr_to_rgb();
    put_data();
    return 0;
}

int calculate(int a, int b);//バイト数計算
void get_data() {
```

```
FILE* fp;

char filename[MAX];
int c;
int filesize, offset, bite_px;//画像の属性


printf("ファイル名を入力して下さい:");
scanf("%s", filename);
fp = fopen(filename, "rb");
if (fp == NULL) {
    printf("%s をオーブンできません¥n", filename);
    exit(1);
}
printf("%s をオーブンしました.¥n", filename);
for (int i = 0; i < 54; i++) {
    c = fgetc(fp);
    header[i] = c;
}
/* printf("¥n¥n<ファイルタイプ>¥n");
for (int i = 0; i < 2; i++) {
  printf("header[%d]=%02x ", i, header[i]);
  } */
printf("¥n¥n ファイルサイズ>¥n");
filesize = calculate(5, 2);
/*  for (int i = 2; i < 6; i++) {
      printf("header[%d]=%02x ", i, header[i]);
  }*/
printf("¥n¥n%d バイト", filesize);
/* printf("¥n¥n<予約領域>¥n");
for (int i = 6; i < 10; i++) {
  printf("header[%d]=%02x ", i, header[i]);
  } */
printf("¥n¥n<オフセット>¥n");
/*for (int i = 10; i < 14; i++) {
    printf("header[%d]=%02x ", i, header[i]);

}*/
offset = calculate(13, 10);
printf("¥n¥n%d バイト", offset);
/*  printf("¥n¥n<情報ヘッダサイズ>¥n");
for (int i = 14; i < 18; i++) {
  printf("header[%d]=%02x ", i, header[i]);
  } */
printf("¥n¥n<画像の幅>¥n");
width = calculate(21, 18);
/*for (int i = 18; i < 22; i++) {
    printf("header[%d]=%02x ", i, header[i]);

}*/
printf("¥n%d 画素¥n", width);
printf("¥n¥n<画像の高さ>");
height = calculate(25, 22);
/*for (int i = 22; i < 26; i++) {
    printf("header[%d]=%02x ", i, header[i]);
}*/
```

```
    printf("¥n%d ライン¥n", height);
    /*  printf("¥n¥n<色プレーン数>¥n");
    for (int i = 26; i < 28; i++) {
      printf("header[%d]=%02x ", i, header[i]);
      } */
    printf("¥n¥n<1 画素にあたりのビット数>¥n");
    bite_px = calculate(29, 28);
    /*  for (int i = 28; i < 30; i++) {
          printf("header[%d]=%02x ", i, header[i]);

      }*/
    printf("¥n%d ビット", bite_px);
    /* printf("¥n¥n<圧縮方式>¥n");
    for (int i = 30; i < 34; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<画像データサイズ>¥n");
    for (int i = 34; i < 38; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<水平解像度>¥n");
    for (int i = 39; i < 42; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<垂直解像度>¥n");
    for (int i = 42; i < 46; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<色数>¥n");
    for (int i = 46; i < 50; i++) {
      printf("header[%d]=%02x ", i, header[i]);
    }
    printf("¥n¥n<重要な色数>¥n");
    for (int i = 50; i < 54; i++) {
      printf("header[%d]=%02x ", i, header[i]);
      } */
    printf("¥n¥n<挿入ビット数>¥n");
    bits = (offset + width * height * (bite_px / 8)) % 4;
    printf("%d バイト¥n", bits);
    //imgin[][][]の初期化
    for (int h = height - 1; h >= 0; h--) {
        for (int w = 0; w < width; w++) {
            for (int i = 2; i >= 0; i--) {
                imgin[i][w][h] = (unsigned char)fgetc(fp);
            }
        }
    }

    fclose(fp);
    printf("¥n%s をクローズしました.¥n", filename);

}
int calculate(int a, int b) {
    int value;
    value = header[a];
    for (int i = a - 1; i >= b; i--) {
        value <<= 8;
```

```
        value += header[i];
    }
    return value;

}

void processing() {

    /* for (int y=height-1;y>=0;y--){
   for(int x=0;x<width;x++){
     for(int i=3;i>0;i--){
       imgout[i][x][y]=imgin[i][width-1-x][y];
     }
   }
 }*/

    /*int i, x, y;
    int copy[3];

    printf("¥n コピーモードを入力して下さい.¥n");
    printf("(コピーする場合 : 1, 固定値に置き換える場合 : 0)¥n");
    for (i = 0; i < 3; i++) {
        printf("%-2s : ", ybr_name[i]);
        scanf("%d", &copy[i]);
    }

    for (i = 0; i < 3; i++) {
        if (copy[i] == 1) {
            for (y = 0; y < height; y++) {
                for (x = 0; x < width; x++) {
                    imgout[i][x][y] = imgin[i][x][y];
                }
            }
        }
        else {
            for (y = 0; y < height; y++) {
                for (x = 0; x < width; x++) {
                    imgout[i][x][y] = 128;
                }
            }
        }
    }
*/
    int i, x, y;

    int block, b_height, b_width;

    block = 2;
    b_height = height / block;
    b_width = width / block;

    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {
                if ((x / b_width + y / b_height) % 2 == 0) {
                    imgout[i][x][y] = imgin[i][x][y];
                }
```

```
                else if (i != 0) {
                    imgout[i][x][y] = 128;
                }
                else {
                    imgout[i][x][y] = 0;
                }
            }
        }
    }
    printf("出力画像データを作成しました.¥n");
}
void put_data() {
    FILE* fp;
    char filename1[MAX];
    int i, x, y;

    printf("出力ファイル名を入力して下さい:");
    scanf("%s", filename1);

    fp = fopen(filename1, "wp");

    printf("%s をオープンしました.¥n", filename1);

    for (i = 0; i < 54; i++) {
        fputc(header[i], fp);
    }


    for (y = height - 1; y >= 0; y--) {
        for (x = 0; x < width; x++) {
            for (i = 2; i >= 0; i--) {
                fputc(imgout[i][x][y], fp);
            }
        }
    }

    for (i = 0; i < bits; i++) {
        fputc('¥0', fp);
    }

    fclose(fp);

    printf("%s をクローズしました.¥n", filename1);

}

void rgb_to_ybr() {
    int i, x, y;
    /*if (height <= 16 || width <= 16) {

        printf("¥n<R 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }
```

```
    printf("¥n<G 信号>¥n");
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            printf("%02x ", imgin[1][x][y]);
        }
        printf("¥n");
    }

    printf("¥n<B 信号>¥n");
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            printf("%02x ", imgin[2][x][y]);
        }
        printf("¥n");
    }
}
else {
    printf("画像サイズが大きいため表示しません.¥n");
}*/

for (y = 0; y < height; y++) {
    for (x = 0; x < width; x++) {

        for (i = 0; i < 3; i++) {
            dtemp[i][x][y] = 0.0;
            for (int j = 0; j < 3; j++)
                dtemp[i][x][y] += rgb_con_ybr[i][j] * (double)imgin[j][x][y];
        }
    }
}

for (y = 0; y < height; y++) {
    for (x = 0; x < width; x++) {
        for (i = 0; i < 3; i++) {

            if (dtemp[i][x][y] > 0.0) {
                itemp[i][x][y] = (int)(dtemp[i][x][y] + 0.5);
            }
            else {
                itemp[i][x][y] = (int)(dtemp[i][x][y] - 0.5);
            }

            if (i != 0) {
                itemp[i][x][y] += 128;
            }

            if (itemp[i][x][y] > 255) {
                itemp[i][x][y] = 255;
            }
            else if (itemp[i][x][y] < 0) {
                itemp[i][x][y] = 0;
            }

            imgin[i][x][y] = (unsigned char)itemp[i][x][y];
        }
    }
```

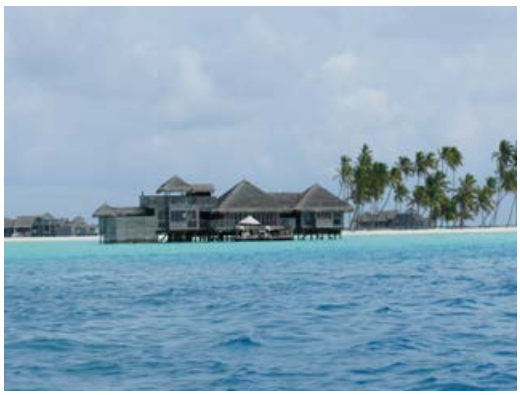
```
    }

    /*printf("¥n<入力 YCbCr 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<Y 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cb 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cr 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

}

void ybr_to_rgb() {
    int i, x, y, j;

   /* printf("¥n<入力 YCbCr 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {

        printf("¥n<Y 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<Cb 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[1][x][y]);
            }
            printf("¥n");
```

```
        }

        printf("¥n<Cr 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgin[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {
                dtemp[i][x][y] = 0.0;
                for (j = 0; j < 3; j++)
                    if (j == 0)
                        dtemp[i][x][y]        +=        ybr_con_rgb[i][j]        *
(double)imgin[j][x][y];
                    else
                        dtemp[i][x][y]        +=        ybr_con_rgb[i][j]        *
(double)(imgin[j][x][y] - 128);
            }
        }
    }


    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            for (i = 0; i < 3; i++) {

                if (dtemp[i][x][y] > 0.0) {
                    itemp[i][x][y] = (int)(dtemp[i][x][y] + 0.5);
                }
                else {
                    itemp[i][x][y] = (int)(dtemp[i][x][y] - 0.5);
                }

                if (itemp[i][x][y] > 255) {
                    itemp[i][x][y] = 255;
                }
                else if (itemp[i][x][y] < 0) {
                    itemp[i][x][y] = 0;
                }

                imgout[i][x][y] = (unsigned char)itemp[i][x][y];
            }
        }
    }

    /*printf("¥n<入力 RGB 信号(整数値)>¥n");

    if (height <= 16 || width <= 16) {
```

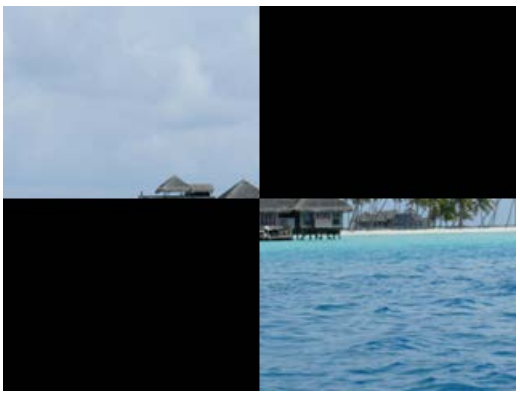
```
        printf("¥n<R 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[0][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<G 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[1][x][y]);
            }
            printf("¥n");
        }

        printf("¥n<B 信号>¥n");
        for (y = 0; y < height; y++) {
            for (x = 0; x < width; x++) {
                printf("%02x ", imgout[2][x][y]);
            }
            printf("¥n");
        }
    }
    else {
        printf("画像サイズが大きいため表示しません.¥n");
    }*/

}
u20216187@gw[32]: gcc -Wall 6-B-2.c -o 6-B-2

u20216187@gw[33]: 6-B-2□□□□□□□[1@.□[1@/

ファイル名を入力して下さい:maldives.bmp
maldives.bmp をオープンしました.


ファイルサイズ>


230456 バイト

<オフセット>


54 バイト

<画像の幅>

320 画素


<画像の高さ>
240 ライン


<1 画素にあたりのビット数>
```

```
24 ビット

<挿入ビット数>
2 バイト

maldives.bmp をクローズしました.
出力画像データを作成しました.
出力ファイル名を入力して下さい:maldives_06_B_2.bmp
maldives_06_B_2.bmp をオープンしました.
maldives_06_B_2.bmp をクローズしました.
u20216187@gw[34]: exit

exit

Script done on Thu Nov  3 14:29:43 2022
```

［添付図］



Maldives.bmp



Maldib¥ves_06_B_2.bmp

【問題アイテムを選択してください。-アイテムを選択してください。】

［プログラムリスト・コンパイル結果・実行結果］

ここをクリックまたはタップしてテキストを入力してください。

［添付図］

ここをクリックまたはタップしてテキストを入力してください。

ここをクリックまたはタップしてテキストを入力してください。