Exercise 9: Recursion, bitwise operations, binary file

9.1 Recursion: Greatest common divisor

Write a program to find the greatest common divisor of two positive integers using recursion

9.2 Recursion: Flip numbers

Write a program which inverts the digits of an integer positive number (ex.: 4325→5234) using recursion.

9.3 Bitwise operation: Count bits

Write a program which asks the user to enter an integer and counts the total number of ones (1s) and zeros (0s) in the given number using bitwise operators.

9.4 Bitwise operation: Unpack data

A particle detector consists of an array of 256x256 pixels (X,Y coordinates) and stores positions and time of each recorded particle in a unique unsigned integer (32 bits) where the first 16 bits store the time, the following 8 bits the Y coordinate and the last 8 bits the X coordinate. Ex.: the event registered at coordinates X = 120, Y = 201 and time t = 4569 is stored as 299485560 whose bit content is 0001000111011001 11001001 01111000 (with 0001000111011001 being 4569 in decimal numbers, 11001001 being 201 and 01111000 being 120)

Write a program which reads the data file det_data.txt then transforms the recorded data in X,Y,t and calculates the mean value of X,Y coordinates and time t.

9.5 Binary file: Write and read binary files

Write a program to convert the above dat_data.txt into a binary file. First 4 bytes should be "unsigned int" for the number of data, the following bytes are the pixel data. Check the file size of the produced file. (It should be 4 bytes + 4 bytes + ndata).

Write a program to read the file and check the data is correctly stored and readable.

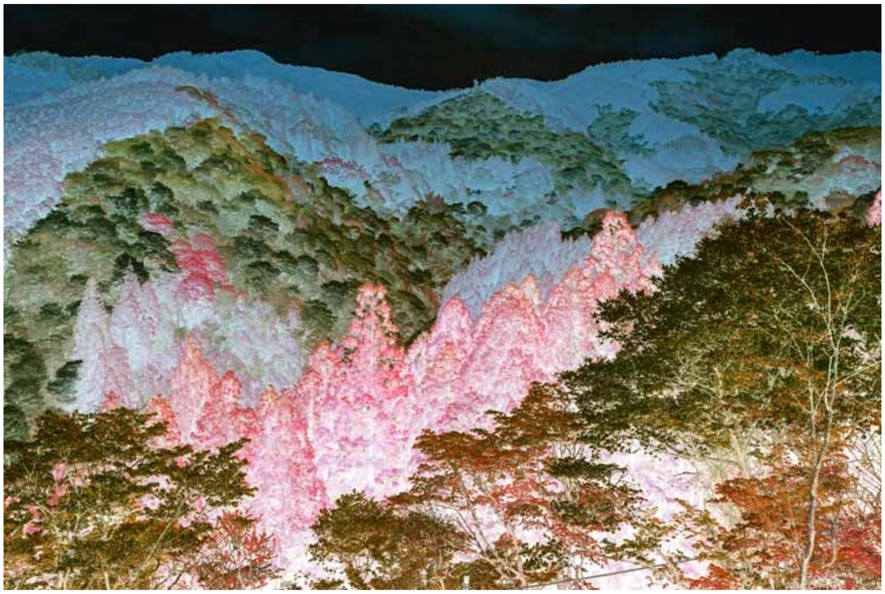

Figure 1: The problematic photo.

9.6 Advanced example (optional): Bitmap operation

I took a photograph of Japanese mountains in April. But, I made a mistake to invert colors, and it seems 2 out of 3 colors (red, green, blue) were swapped in some way. Please try to fix it.

The bitmap has 54 bytes of the BmpHeader, and then data follows. The color data consists of 3 bytes (1 byte for each of red, green blue). Each color data ranges 0-255 (or 0x0 - 0xff). Example code to read and write new bmp file is given.

```
#include<stdio.h>
#pragma pack(2) // defines how to map the data bytes in a structure

typedef struct
{
    // Bitmap header structure
    char ID[2];
    long FileSize;
    long RESERVED;
    long BitmapDataOffset;
    long BitmapHeaderSize;
    long Width;
    long Height;
    short NrOfPlanes;
    short BitsPerPixel;
    long Compression;
    long BitmapDataSize;
    long HorizontalResolution;
    long VerticalResolution;
    long Colors;
    long ImportantColors;
} BmpHeader;

int main(){
  // color data
  unsigned char c1[3], c2[3];
```

```c
  // Open input file with read-binary mode
  FILE *fp = fopen("sakura.mod.bmp","rb");
  BmpHeader h;

  // Read headder
  fread(&h, sizeof(BmpHeader), 1, fp);
  printf("Width = %d, Height = %d, %d bits per pixel\n", h.Width, h.Height
   , h.BitsPerPixel);
  printf("Offset To data %d\n", h.BitmapDataOffset);

  // Open output file with write-binary mode, and write the BMP header.
  FILE *fpo = fopen("sakura.new.bmp", "wb");
  fwrite(&h, sizeof(BmpHeader), 1, fpo);

  // move file pointer to the data position
  fseek(fp, h.BitmapDataOffset, 0);

  // loop over all the pixels
  for(int iy=0; iy<h.Height; iy++){
    for(int ix=0; ix<h.Width; ix++){
      // read 1x3 bytes from input file
      fread(c1, sizeof(unsigned char), 3, fp);

      // test output
      if(iy==0) printf("%x%x%x ", 0xff&c1[0], 0xff&c1[1], 0xff&c1[2]);

      // Implement something here.
      // (as an example, the below is copying c1 to c2).
      c2[0] = c1[0];
      c2[1] = c1[1];
      c2[2] = c1[2];

      // write 1x3 bytes into output file
      fwrite(&c2, sizeof(unsigned char), 3, fpo);
    }
  }

  fclose(fp);
  fclose(fpo);

}
```