중간 프로젝트 과제1

201920952_김준영

2. 자동차 번호판(또는 주민등록증, 운전면허증, 여권, 학생증, 사원증 등등)에서 번호 숫자들을 각각 직사각형으로 둘러싸고, 좌표(x1, y1)-(x2, y2)를 출력하게 하라

(c) 소스 코드

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    char M, N;
    int XX, YY, MAX, y, x;
    FILE *file1, *file2;
    int image1[1200][1000];

    file1 = fopen(argv[1], "r"); // 파일 읽기.
    fscanf(file1, "%c%c", &M, &N);
    fscanf(file1, "%d%d", &XX, &YY);
    fscanf(file1, "%d", &MAX);
    for (y = 0; y < YY; y++)
        for (x = 0; x < XX; x++)
            fscanf(file1, "%d", &image1[y][x]);

   for(y = 0; y < YY; y++) // 이진화.
    for(x = 0; x < XX; x++)
     if(image1[y][x] < 100) // 100이하라면 0, 그 이상이면 255
        image1[y][x] = 0;
     else
        image1[y][x] = 255;

    // y값 초기화
    int y1 = YY;
    int y2 = -1;

    // 최대, 최소값으로 숫자를 감싸는 위,아래 값 구하기
    for (y = 0; y < YY; y++)
        for (x = 0; x < XX; x++)
            if (image1[y][x] == 0)
            {
                if (y < y1) y1 = y;
```

```
                if (y > y2) y2 = y;
            }

    int x1[10], x2[10]; // x1, x2 시작과 끝 좌표
    int numIndex = 0; // x1,x2의 인덱스
    int inBlack = 0; // 검은색 픽셀 영역에 있는지 확인여부.

    // x1, x2값 구하기
    for (x = 0; x < XX; x++)
    {
        int blockC = 0;
        for (y = y1; y <= y2; y++)
            if (image1[y][x] == 0) // 검은색(픽셀)을 만난다면
            {
                blockC = 1; // 검은색을 만났다는 flag
                if (inBlack == 0)
                {
                    inBlack = 1; // 검은색 영역에 있다.
                    x1[numIndex] = x;
                }
            }

        // 검은색을 만나지 않았고 이전에 검은색 영역이었다면 x2값 넣기
        if (blockC == 0 && inBlack == 1)
        {
            x2[numIndex] = x - 1;
            numIndex++; // 인덱스 증가. 다음 숫자를 담기 위함
            inBlack = 0; // 검은색 영역에 x. 초기화
        }
    }

    for (int num = 0; num < numIndex; num++)
    {
        for (y = y1; y <= y2; y++) // 세로축 그리기
        {
            image1[y][x1[num]] = 0;
            image1[y][x2[num]] = 0;
        }
        for (x = x1[num]; x <= x2[num]; x++) // 가로축 그리기
        {
```

```
                image1[y1][x] = 0;
                image1[y2][x] = 0;
            }
        }

        file2 = fopen(argv[2], "w"); // 파일 쓰기
        fprintf(file2, "%c%c\n", M, N);
        fprintf(file2, "%d %d\n", XX, YY);
        fprintf(file2, "%d\n", MAX);

        for (y = 0; y < YY; y++) //
        {
            for (x = 0; x < XX; x++)
                fprintf(file2, "%4d", image1[y][x]);
            fprintf(file2, "\n");
        }

        // (x1, y1) - (x2, y2)의 값을 출력
        printf("\n");
        for (int num = 0; num < numIndex; num++)
        {
            printf("%2d번째 숫자 : (x1=%3d, y1=%d) - (x2=%3d, y2=%d) = (%d, %d)\n",
num + 1, x1[num], y1, x2[num], y2, x2[num] - x1[num] , y2 - y1);
        }

        return 0;
}
```

입력영상(기울어짐 현상)

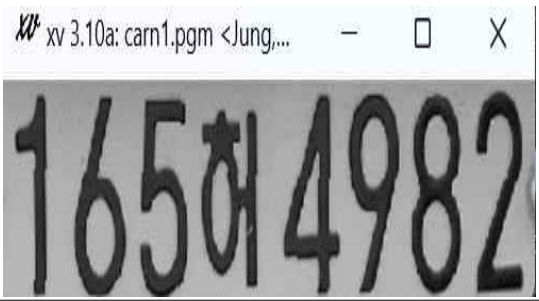
수평으로 맞추기

```
#include <opencv2/opencv.hpp>
using namespace cv;
int main(int argc, char *argv[])
{
  Mat image1 = imread(argv[1], IMREAD_GRAYSCALE); // 그레이스케일로 읽기
  Mat image2 = Mat();

  Point center = Point(image1.cols/2.0, image1.rows/2.0);
  Mat M = getRotationMatrix2D(center, -16.730, 1); // 각도 조절
  warpAffine(image1, image2, M, Size(image1.cols, image1.rows), 1);
  imshow("input", image1);
  imshow("output", image2);
  waitKey(0);
  return 0;
}
// atan((171-275)/(679-333)) * 180/pi = -16.730
```

입력영상1

입력영상2



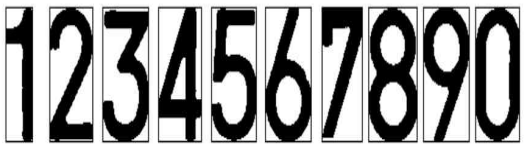
1. 명령어 화면

```
[kimjunyoung@raspberry4 ~]nano mynum.c
[kimjunyoung@raspberry4 ~]gcc mynum.c
[kimjunyoung@raspberry4 ~]a.out /image/B1234567890.pgm o.pgm &
[4] 27110
[kimjunyoung@raspberry4 ~]
 1번 째  숫 자  : (x1= 18, y1=100) - (x2= 47, y2=187) = (29, 87)
 2번 째  숫 자  : (x1= 66, y1=100) - (x2=113, y2=187) = (47, 87)
 3번 째  숫 자  : (x1=124, y1=100) - (x2=174, y2=187) = (50, 87)
 4번 째  숫 자  : (x1=185, y1=100) - (x2=232, y2=187) = (47, 87)
 5번 째  숫 자  : (x1=243, y1=100) - (x2=292, y2=187) = (49, 87)
 6번 째  숫 자  : (x1=302, y1=100) - (x2=352, y2=187) = (50, 87)
 7번 째  숫 자  : (x1=364, y1=100) - (x2=408, y2=187) = (44, 87)
 8번 째  숫 자  : (x1=415, y1=100) - (x2=468, y2=187) = (53, 87)
 9번 째  숫 자  : (x1=475, y1=100) - (x2=525, y2=187) = (50, 87)
10번 째  숫 자  : (x1=532, y1=100) - (x2=580, y2=187) = (48, 87)
```

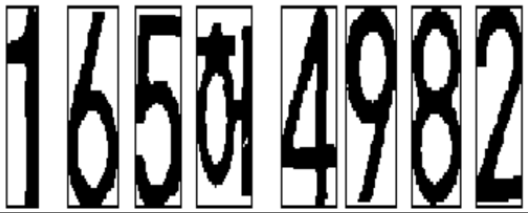
2. 명령어 화면

```
[kimjunyoung@raspberry4 ~]nano mynum.c
[kimjunyoung@raspberry4 ~]gcc mynum.c
[kimjunyoung@raspberry4 ~]a.out carn1.pgm o.pgm &
[4] 27051
[kimjunyoung@raspberry4 ~]
 1번 째  숫 자  : (x1=  8, y1=4) - (x2= 27, y2=72) = (19, 68)
 2번 째  숫 자  : (x1= 46, y1=4) - (x2= 77, y2=72) = (31, 68)
 3번 째  숫 자  : (x1= 87, y1=4) - (x2=115, y2=72) = (28, 68)
 4번 째  숫 자  : (x1=125, y1=4) - (x2=159, y2=72) = (34, 68)
 5번 째  숫 자  : (x1=177, y1=4) - (x2=211, y2=72) = (34, 68)
 6번 째  숫 자  : (x1=217, y1=4) - (x2=248, y2=72) = (31, 68)
 7번 째  숫 자  : (x1=257, y1=4) - (x2=287, y2=72) = (30, 68)
 8번 째  숫 자  : (x1=297, y1=4) - (x2=325, y2=72) = (28, 68)
```

출력영상1



출력영상2



코드 설명

번호판 영상이 있을 때, 각각의 숫자들을 직사각형으로 둘러싸고 (x1, y1) - (x2, y2) 값을 출력한다. 먼저 이 코드의 알고리즘은 pgm으로 된 번호판을 이진화 하였기 때문에 0(검은색)과 255(흰색)의 픽셀로 이루어져 있다. y1, y2를 최대, 최소값으로 맨위의 픽셀값과 맨 아래의 픽셀 값을 구한다. x1, x2는 이중반복문으로 y값으로 내려가며 0(검은색)을 만난다면 x1에 값을 넣고 계속 반복하다가 검은색을 만나지 않는다면 숫자의 끝을 의미하므로 x2를 구할 수 있다.

crop 방법으로 template Mating

1. 크기 정규화를 한다.
2. make_temp 배열값을 만들어준다.
3. Hamming Distance로 비교해준다.

1. 크기 정규화(normal size)

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  char M, N;
  int XX ,YY, MAX, x, y, xx, yy;
  FILE *file1, *file2;
  int image1[1200][1000];
  int image2[1200][1000];

  file1 = fopen(argv[1], "r");
```

```
  fscanf(file1, "%c%c",&M,&N);
  fscanf(file1, "%d%d",&XX,&YY);
  fscanf(file1, "%d",&MAX);
  for(y = 0; y < YY; y++)
   for(x = 0; x < XX; x++)
    fscanf(file1, "%d", &image1[y][x]);

  for(y = 0; y < YY; y++) // 20 x 30으로 크기 정규화
    for(x = 0; x < XX; x++)
    {
      yy = y * 30/YY;
      xx = x * 20/XX;
      image2[yy][xx] = image1[y][x];
    }

  XX = XX * 20/XX;
  YY = YY * 30/YY;

  file2 = fopen(argv[2], "w");
  fprintf(file2,"%c%c\n", M, N);
  fprintf(file2,"%d %d\n", XX, YY);
  fprintf(file2,"%d\n", MAX);

  for(y = 0; y < YY; y++)
  {
   for(x = 0; x < XX; x++)
   {
        fprintf(file2, "%4d", image2[y][x]);
   }
   fprintf(file2,"\n");
  }
  return 0;
}
```

2. make_temp (배열 생성) - 헤더에 쓰기 위함

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  char M, N;
  int XX ,YY, MAX, x, y, xx, yy;
  FILE *file1, *file2;
```

```
  int image[600][800];

  file1 = fopen(argv[1], "r");
  fscanf(file1, "%c%c",&M,&N);
  fscanf(file1, "%d%d",&XX,&YY);
  fscanf(file1, "%d",&MAX);
  for(y = 0; y < YY; y++)
   for(x = 0; x < XX; x++)
    fscanf(file1, "%d", &image[y][x]);


  printf("int [30][20]={\n");

  for(y = 0; y < YY; y++)
  {
   for(x = 0; x < XX; x++)
   {
      printf("%4d, ", image[y][x]);
   }
      printf("\n");
  }
  return 0;
}
```

2-1. make_temp T1N.pgm > template1.h (이름과 배열 괄호를 확인한다)

3. Hamming Distance로 확인

```
#include <stdio.h>
#include <stdlib.h>
#include "tem0.h"
#include "tem1.h"
#include "tem2.h"
#include "tem5.h"
#include "tem8.h"
int main(int argc, char *argv[])
{
  char M, N;
  int XX ,YY, MAX, x, y, xx, yy;
  FILE *file1, *file2;
  int image1[1200][1000];
  int image2[1200][1000];

  int distance[5] = {0};
```

```
int i;
int class,  min = 600;

file1 = fopen(argv[1], "r");
fscanf(file1, "%c%c",&M,&N);
fscanf(file1, "%d%d",&XX,&YY);
fscanf(file1, "%d",&MAX);

for(y = 0; y < YY; y++)
 for(x = 0; x < XX; x++)
  fscanf(file1, "%d", &image1[y][x]);

for(y = 0; y < YY; y++)
 for(x = 0; x < XX; x++)
 {
    if(abs(image1[y][x] - zero[y][x]) == 255)
       distance[0]++;
    if(abs(image1[y][x] - one[y][x]) == 255)
       distance[1]++;
    if(abs(image1[y][x] - two[y][x]) == 255)
       distance[2]++;
    if(abs(image1[y][x] - five[y][x]) == 255)
       distance[3]++;
    if(abs(image1[y][x] - eight[y][x]) == 255)
       distance[4]++;


 }

for(i = 0; i < 5; i++)
 printf("distance[%d] = %d\n", i, distance[i]);

for(i = 0; i < 5; i++)
 if(distance[i] < min)
 {
  min = distance[i];
  class = i;
 }

printf("The input character is %d\n", class);
```

```
    return 0;
}
```

입력 영상

이진화 영상

4. 숫자 인식 확인

1) 숫자 0 인식

```
[kimjunyoung@raspberry4 ~]gcc template3.c
[kimjunyoung@raspberry4 ~]a.out hw0N.pgm
distance[0] = 0
distance[1] = 210
distance[2] = 203
distance[3] = 219
distance[4] = 186
The input character is 0
```

2) 숫자 1 인식

```
[kimjunyoung@raspberry4 ~]gcc template3.c
[kimjunyoung@raspberry4 ~]a.out hw1N.pgm
distance[0] = 210
distance[1] = 0
distance[2] = 245
distance[3] = 287
distance[4] = 282
The input character is 1
```

3) 숫자 2 인식

```
[kimjunyoung@raspberry4 ~]gcc template3.c
[kimjunyoung@raspberry4 ~]a.out hw2N.pgm
distance[0] = 203
distance[1] = 245
distance[2] = 0
distance[3] = 268
distance[4] = 291
The input character is 2
```

4) 숫자 5 인식

```
[kimjunyoung@raspberry4 ~]gcc template3.c
[kimjunyoung@raspberry4 ~]a.out hw5N.pgm
distance[0] = 219
distance[1] = 287
distance[2] = 268
distance[3] = 0
distance[4] = 221
The input character is 3
```

4) 숫자 8 인식

```
[kimjunyoung@raspberry4 ~]gcc template3.c
[kimjunyoung@raspberry4 ~]a.out hw8N.pgm
distance[0] = 186
distance[1] = 282
distance[2] = 291
distance[3] = 221
distance[4] = 0
The input character is 4
```

각각의 숫자를 출력하는 코드.

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    char M, N;
    int XX, YY, MAX, y, x;
    FILE *file[11];
```

```
    int image1[1200][1000];

    file[0] = fopen(argv[1], "r");
    fscanf(file[0], "%c%c", &M, &N);
    fscanf(file[0], "%d%d", &XX, &YY);
    fscanf(file[0], "%d", &MAX);
    for (y = 0; y < YY; y++)
        for (x = 0; x < XX; x++)
            fscanf(file[0], "%d", &image1[y][x]);

  for(y = 0; y < YY; y++)
    for(x = 0; x < XX; x++)
     if(image1[y][x] < 100)
        image1[y][x] = 0;
     else
        image1[y][x] = 255;

    int y1 = YY; // 초기화
    int y2 = -1;

    for (y = 0; y < YY; y++)
        for (x = 0; x < XX; x++)
            if (image1[y][x] == 0)
            {
                if (y < y1) y1 = y;
                if (y > y2) y2 = y;
            }

    int x1[10], x2[10];
    int numIndex = 0;
    int inBlack = 0;

    for (x = 0; x < XX; x++)
    {
        int blockC = 0;
        for (y = y1; y <= y2; y++)
            if (image1[y][x] == 0)
            {
                blockC = 1;
                if (inBlack == 0)
```

```
                {
                    inBlack = 1;
                    x1[numIndex] = x;
                }
            }

        if (blockC == 0 && inBlack == 1)
        {
            x2[numIndex] = x - 1;
            numIndex++;
            inBlack = 0;
        }
    }

    for (int num = 0; num < numIndex; num++)
    {
        for (y = y1; y <= y2; y++)
        {
            image1[y][x1[num]] = 0;
            image1[y][x2[num]] = 0;

        }
        for (x = x1[num]; x <= x2[num]; x++)
        {
            image1[y1][x] = 0;
            image1[y2][x] = 0;
        }
    }


    file[1] = fopen(argv[2],"w");
    fprintf(file[1], "%c%c\n",M,N);
    fprintf(file[1], "%d %d\n",XX,YY);
    fprintf(file[1], "%d\n",MAX);
    for(y = 0; y < YY; y++)
    {
       for(x = 0; x < XX; x++)
      {
        fprintf(file[1],"%4d ",image1[y][x]);
      }
```

```
        fprintf(file[1],"\n");
    }


    for(int i = 0; i < numIndex; i++){
        file[i+2] = fopen(argv[i+2], "w");
        fprintf(file[i+2], "%c%c\n", M, N);
        fprintf(file[i+2], "%d %d\n", x2[i] - x1[i]+1, y2-y1+1);
        fprintf(file[i+2], "%d\n", MAX);
    }

    for(int i = 0; i < numIndex; i++){
     for (y = y1; y <= y2; y++)
     {
        for (x = x1[i]; x <= x2[i]; x++)
            fprintf(file[i+2], "%4d", image1[y][x]);
        fprintf(file[i+2], "\n");
     }
    }

    // x1, y1 및 x2, y2의 값을 출력
    printf("\n");
    for (int num = 0; num < numIndex; num++)
    {
        printf("%2d번째 숫자 : (x1=%3d, y1=%d) - (x2=%3d, y2=%d) = (%d, %d)\n",
        num + 1, x1[num], y1, x2[num], y2, x2[num] - x1[num], y2 - y1);
    }

    return 0;
}
```