資料結構演算法

與整合程式設計實務

1071 電機系

期末考報告

五角星圖形 BMP 檔

電機三 B

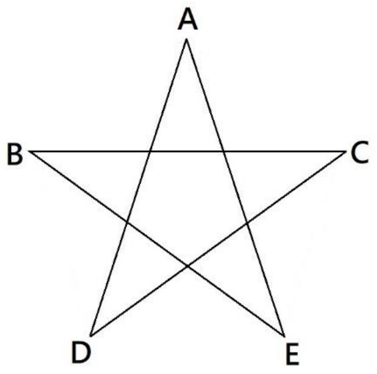
王靖嫻

B1055157

蕭培墉老師

目錄

一、題目解析



由於 BMP 檔圖形資訊儲存在 BMP_Bitmap Data Block 內，所以必須令一陣列 bPixel []代表 BMP_Bitmap Data Block 來儲存這些點構成的五條線段($\overline{\mathrm{AD}}$、$\overline{\mathrm{DC}}$、$\overline{\mathrm{CB}}$、$\overline{\mathrm{BE}}$、$\overline{\mathrm{EA}}$，如右圖)，為了取得五條線段上每個點的準確座標，必須將五個點(A、B、C、D、E)座標先求出，已知 B 和 C 的座標，透過此五角星的幾何性質可將其他三個點座標求出(下節詳述)，當五點都已確定，就可以求出五條線段的線性函數，接下來依序將在線段上的所有點座標以灰階強度 0(正黑，也可使用不同強度數值)填入位於 BMP_Bitmap Data Block 之陣列 bPixel[]中。

需準備一空白 BMP 輸入檔及輸入文字檔，檔名命為 IPBmpFile.txt，內容含

(1) 8bit 輸入 BMP 檔名(Blank.bmp)

(2) 8bit 輸出 BMP 檔名(Draw_Star.bmp)

(3) 8bit 輸入影像寬度

(4) 8bit 輸入影像高度

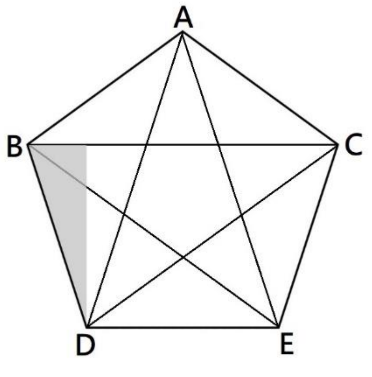
(5) B 點座標

(6) C 點座標

二、解題原理與數學

1. 求出 ADE 點座標

令點 K(K_x , K_y) ，K 可為 A,B,C,D,E。



為了方便觀察五角星的幾何性質，將五端點外圍連成一正五邊形(如圖)，已知 BC 點，首先將 D 點求出，由於五角星之每一內角為 180°/ 5=36°，又∆BCD 為等腰三角形，故∠CBD 為 72°，所以

$D_x= B_x+\overline{BD}*\cos72^\circ$

$D_y= B_y -\overline{BD}*\sin72^\circ$

令等腰三角形之兩腰為 tri_a 變數，底邊為 tri_b 變數

tri_a=C_x－B_x

tri_b= 2*tri_a *cos72°

改寫 D 點座標表示式

$D_x = B_x+$ tri_b *cos72°

$D_y = B_y -$ tri_b *sin72°

$E_x = C_x -$ tri_b *cos72°

$E_y = D_y$

A 點座標為

$A_x= (B_x +C_x)/2$

$A_y=D_y+tri\_a*\sin72°$

為程式撰寫方便，令 A_x ,B_x ,C_x ,D_x ,E_x 對應值為 X[k] , k : 0,1,2,3,4，Y[k]亦然。

另外在 math.h 函式庫中三角函數 sin()、cos()的定義為

double sin(double x)

⇒ returns the sine of a **radian** angle x

double cos(double x)

⇒ returns the cosine of a **radian** angle x

因此使用函式時須將角度轉換為弧度。

2. 求出五條線段的線性函數

Line_AD:

$y=f(x)=m_{AD}*x+b_{AD}$

$m_{AD}= (A_y - D_y)/(A_x - D_x)$

$b_{AD}= A_y- m_{AD}* A_x$

Line_DC:

$y=f(x)=m_{DC}*x+b_{DC}$

$m_{DC}= (D_y - C_y)/(D_x - C_x)$

$b_{DC}= D_y- m_{DC}* D_x$

Line_CB:

$y=f(x)=m_{CB}*x+b_{CB}$

$m_{CB}= (C_y - B_y)/(C_x - B_x)$

$b_{CB}= C_y- m_{CB}* C_x$

Line_BE:

$y=f(x)=m_{BE}*x+b_{BE}$

$m_{BE}= (B_y - E_y)/(B_x - E_x)$

$b_{BE}= B_y- m_{BE}* B_x$

Line_EA:

$y=f(x)=m_{EA}*x+b_{EA}$

$m_{EA}= (E_y - A_y)/(E_x - A_x)$

$b_{EA}= E_y- m_{EA}* E_x$

為程式撰寫方便，令 m_{AD}, m_{DC}, m_{CB}, m_{BE}, m_{EA} 對應值為 Line_m[k] , k : 0,1,2,3,4，Line_b[k]亦然。

三、演算法與流程圖

1. 操作步驟(演算法)

I. 讀取輸入文字檔 IPBmpFile.txt，將輸入 bmp 檔名、輸出 bmp 檔名、輸入 bmp

影像寬度、輸入 bmp 影像高度、B 點座標、C 點座標存入自創資料結構。

II. 利用 fopen()+rb 參數來開啟輸入圖像，並藉由 fseek()、fread()來讀取四大區塊的對應 byte 位置之 16 進位數值，最後將所有數值存入自創資料結構(BMPHeaderData *BMPHeaderData_ptr)。

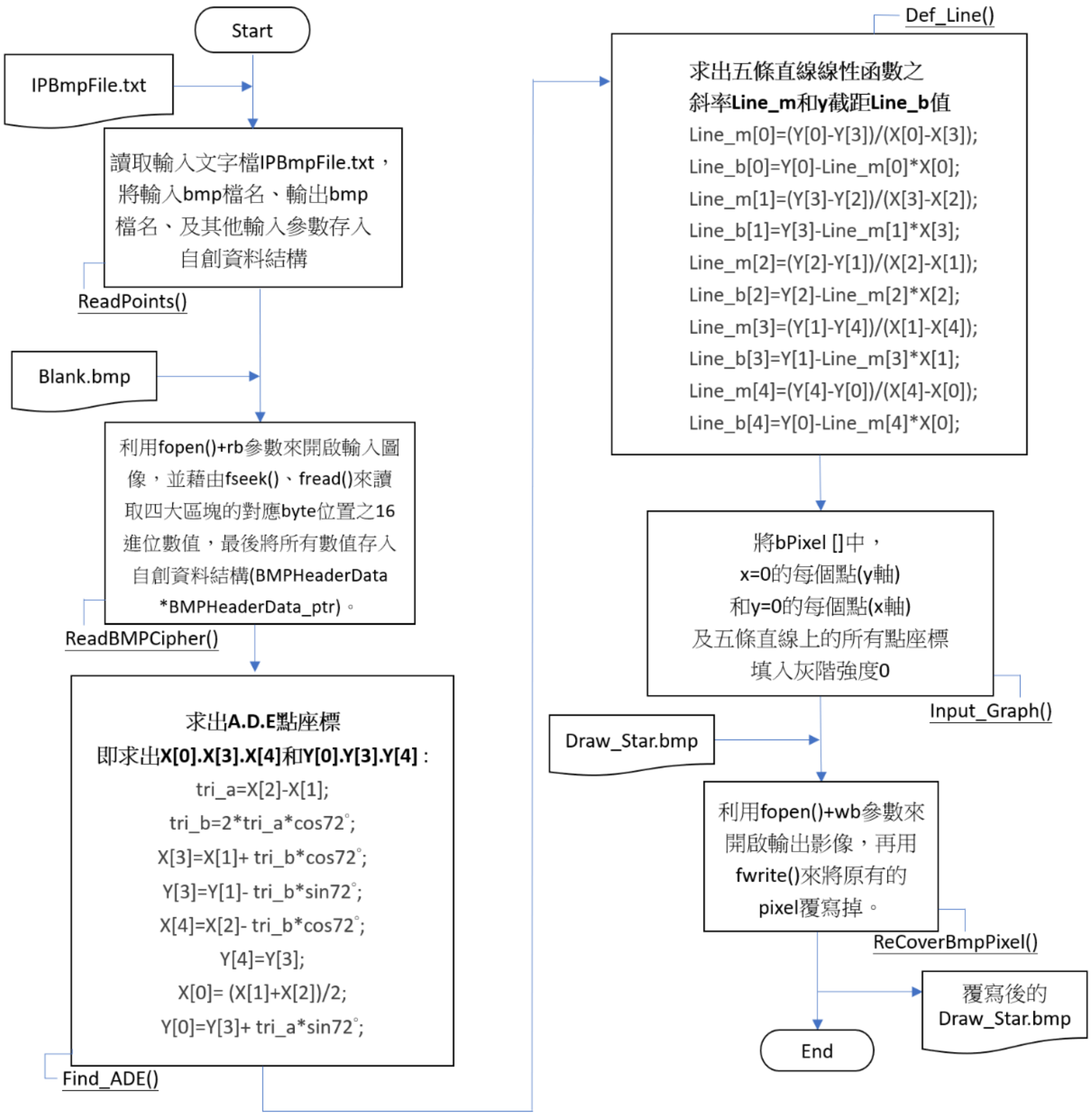
III. 利用上述數學推導求出 A、D、E 點座標。

IV. 利用五端點座標，將其所連成的五條直線之線型函數求出。

V. 將 X 軸、Y 軸、五條直線裡的所有點座標以灰階強度 0(正黑)填入陣列 bPixel[]。

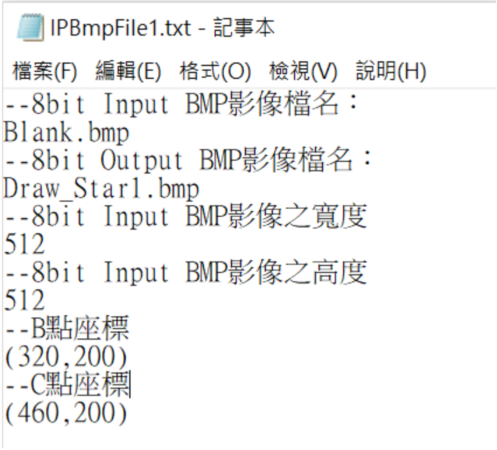
VI. 利用 fopen()+wb 參數來開啟輸出影像，再用 fwrite()來將原有的 pixel 覆寫掉。

2. 流程圖


Start
IPBmpFile.txt
讀取輸入文字檔IPBmpFile.txt，將輸入bmp檔名、輸出bmp檔名、及其他輸入參數存入自創資料結構
ReadPoints()
Blank.bmp
利用fopen()+rb參數來開啟輸入圖像，並藉由fseek()、fread()來讀取四大區塊的對應byte位置之16進位數值，最後將所有數值存入自創資料結構(BMPHeaderData *BMPHeaderData_ptr)。
ReadBMPCipher()
求出A.D.E點座標
即求出X[0].X[3].X[4]和Y[0].Y[3].Y[4]：
tri_a=X[2]-X[1];
tri_b=2*tri_a*cos72°;
X[3]=X[1]+ tri_b*cos72°;
Y[3]=Y[1]- tri_b*sin72°;
X[4]=X[2]- tri_b*cos72°;
Y[4]=Y[3];
X[0]= (X[1]+X[2])/2;
Y[0]=Y[3]+ tri_a*sin72°;
Find_ADE()
Def_Line()
求出五條直線線性函數之斜率Line_m和y截距Line_b值
Line_m[0]=(Y[0]-Y[3])/(X[0]-X[3]);
Line_b[0]=Y[0]-Line_m[0]*X[0];
Line_m[1]=(Y[3]-Y[2])/(X[3]-X[2]);
Line_b[1]=Y[3]-Line_m[1]*X[3];
Line_m[2]=(Y[2]-Y[1])/(X[2]-X[1]);
Line_b[2]=Y[2]-Line_m[2]*X[2];
Line_m[3]=(Y[1]-Y[4])/(X[1]-X[4]);
Line_b[3]=Y[1]-Line_m[3]*X[1];
Line_m[4]=(Y[4]-Y[0])/(X[4]-X[0]);
Line_b[4]=Y[0]-Line_m[4]*X[0];
將bPixel []中，x=0的每個點(y軸)和y=0的每個點(x軸)及五條直線上的所有點座標填入灰階強度0
Input_Graph()
Draw_Star.bmp
利用fopen()+wb參數來開啟輸出影像，再用fwrite()來將原有的pixel覆寫掉。
ReCoverBmpPixel()
覆寫後的Draw_Star.bmp
End

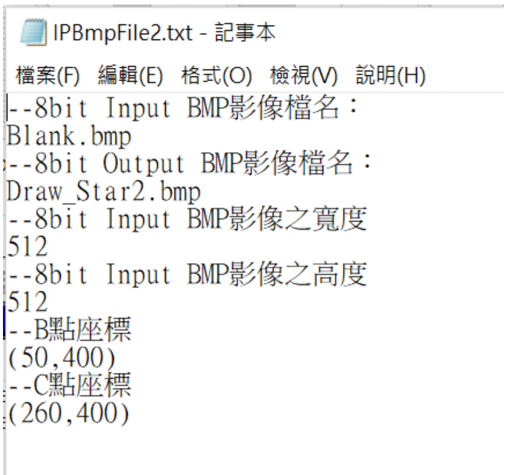
四、輸出入結果

- 輸入文字檔一 IPBmpFile1.txt



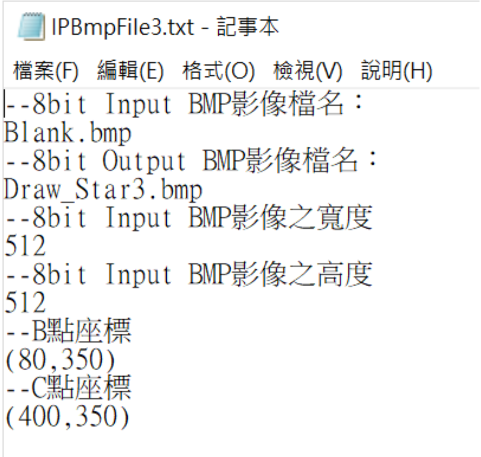
```
--8bit Input BMP影像檔名：
Blank.bmp
--8bit Output BMP影像檔名：
Draw_Star1.bmp
--8bit Input BMP影像之寬度
512
--8bit Input BMP影像之高度
512
--B點座標
(320,200)
--C點座標
(460,200)
```

- 輸入文字檔二 IPBmpFile2.txt



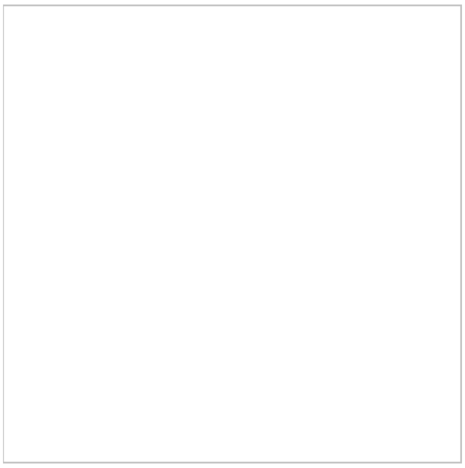
```
--8bit Input BMP影像檔名：
Blank.bmp
--8bit Output BMP影像檔名：
Draw_Star2.bmp
--8bit Input BMP影像之寬度
512
--8bit Input BMP影像之高度
512
--B點座標
(50,400)
--C點座標
(260,400)
```

- 輸入文字檔三 IPBmpFile3.txt

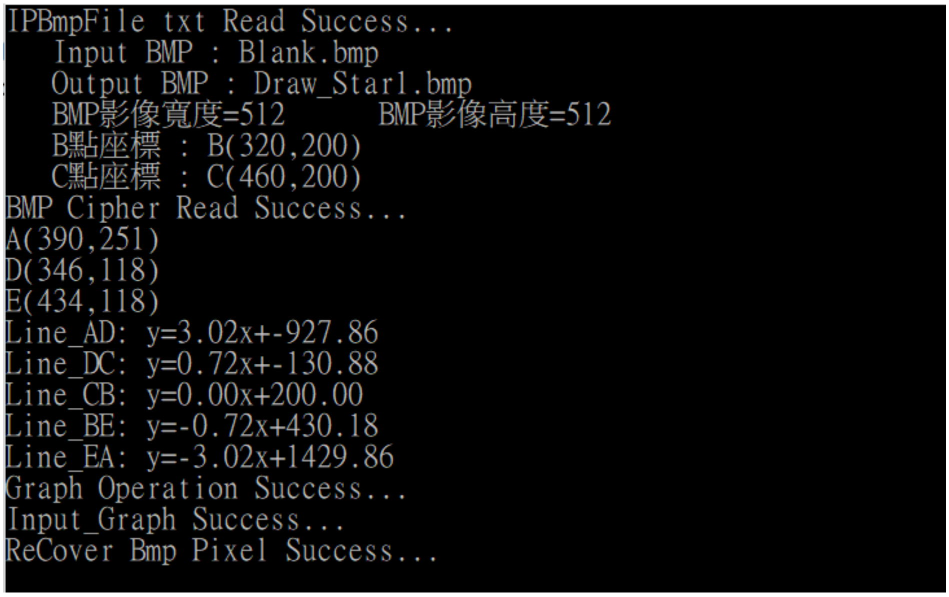


```
--8bit Input BMP影像檔名：
Blank.bmp
--8bit Output BMP影像檔名：
Draw_Star3.bmp
--8bit Input BMP影像之寬度
512
--8bit Input BMP影像之高度
512
--B點座標
(80,350)
--C點座標
(400,350)
```

- 輸入空白 BMP 檔 Blank.bmp

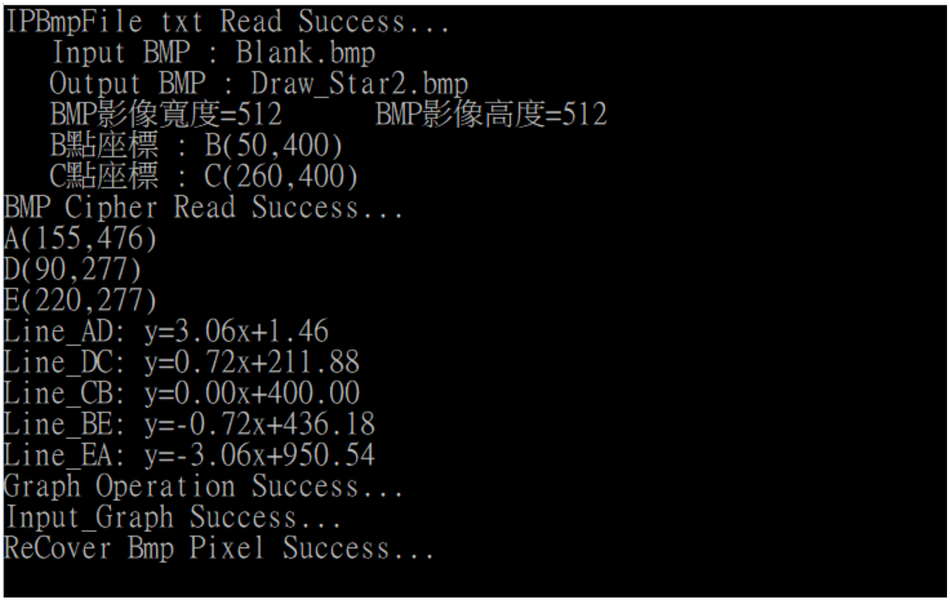


- 執行結果一



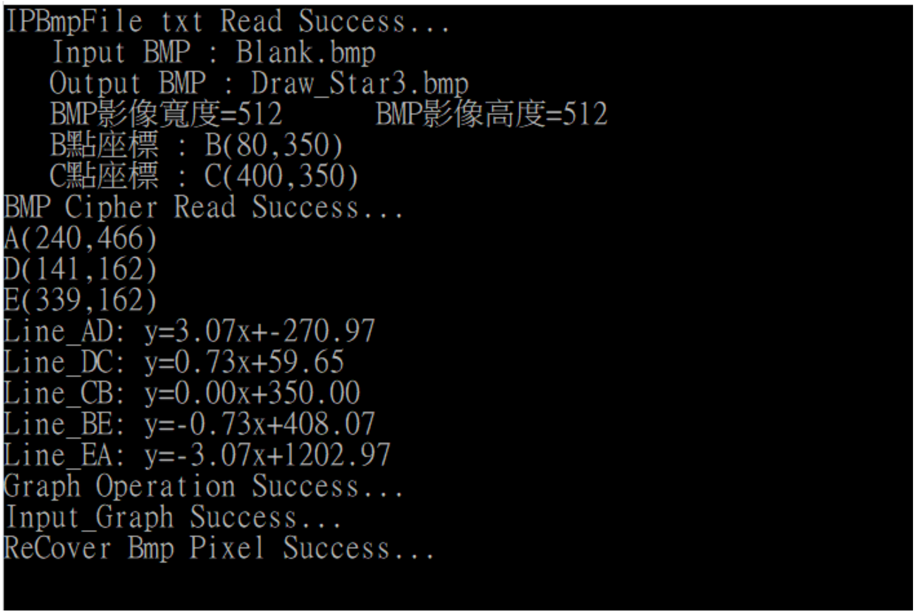
```
IPBmpFile txt Read Success...
   Input BMP : Blank.bmp
   Output BMP : Draw_Star1.bmp
   BMP影像寬度=512      BMP影像高度=512
   B點座標 : B(320,200)
   C點座標 : C(460,200)
BMP Cipher Read Success...
A(390,251)
D(346,118)
E(434,118)
Line_AD: y=3.02x+-927.86
Line_DC: y=0.72x+-130.88
Line_CB: y=0.00x+200.00
Line_BE: y=-0.72x+430.18
Line_EA: y=-3.02x+1429.86
Graph Operation Success...
Input_Graph Success...
ReCover Bmp Pixel Success...
```

- 執行結果二



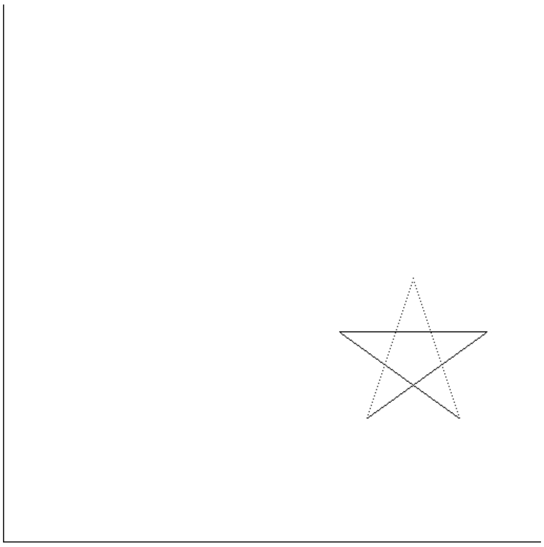
```
IPBmpFile txt Read Success...
   Input BMP : Blank.bmp
   Output BMP : Draw_Star2.bmp
   BMP影像寬度=512      BMP影像高度=512
   B點座標 : B(50,400)
   C點座標 : C(260,400)
BMP Cipher Read Success...
A(155,476)
D(90,277)
E(220,277)
Line_AD: y=3.06x+1.46
Line_DC: y=0.72x+211.88
Line_CB: y=0.00x+400.00
Line_BE: y=-0.72x+436.18
Line_EA: y=-3.06x+950.54
Graph Operation Success...
Input_Graph Success...
ReCover Bmp Pixel Success...
```

- 執行結果三

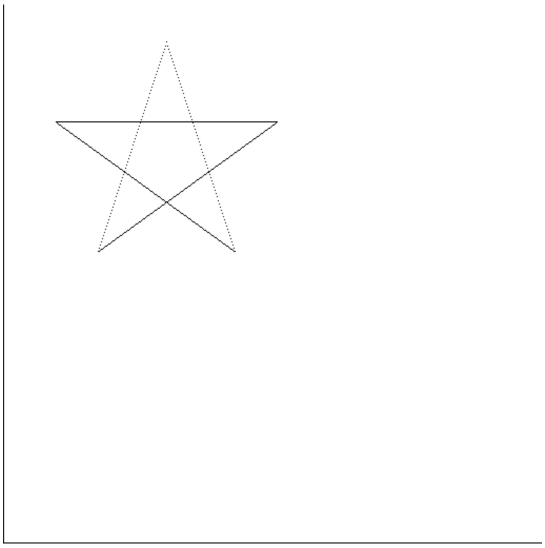


```
IPBmpFile txt Read Success...
   Input BMP : Blank.bmp
   Output BMP : Draw_Star3.bmp
   BMP影像寬度=512      BMP影像高度=512
   B點座標 : B(80,350)
   C點座標 : C(400,350)
BMP Cipher Read Success...
A(240,466)
D(141,162)
E(339,162)
Line_AD: y=3.07x+-270.97
Line_DC: y=0.73x+59.65
Line_CB: y=0.00x+350.00
Line_BE: y=-0.73x+408.07
Line_EA: y=-3.07x+1202.97
Graph Operation Success...
Input_Graph Success...
ReCover Bmp Pixel Success...
```

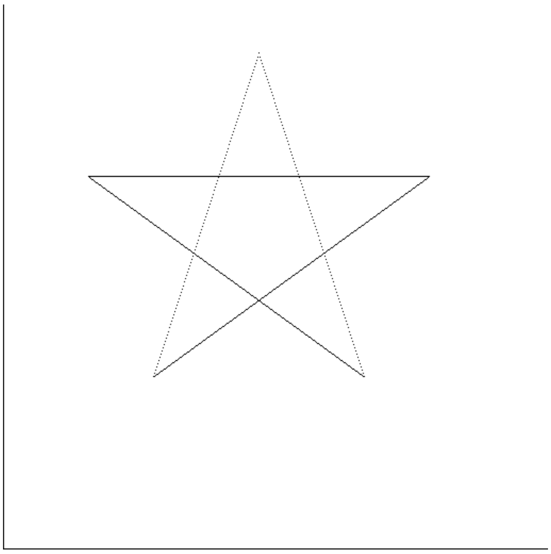
- 輸出文字檔一 Draw_Star1.bmp



- 輸出文字檔二 Draw_Star2.bmp

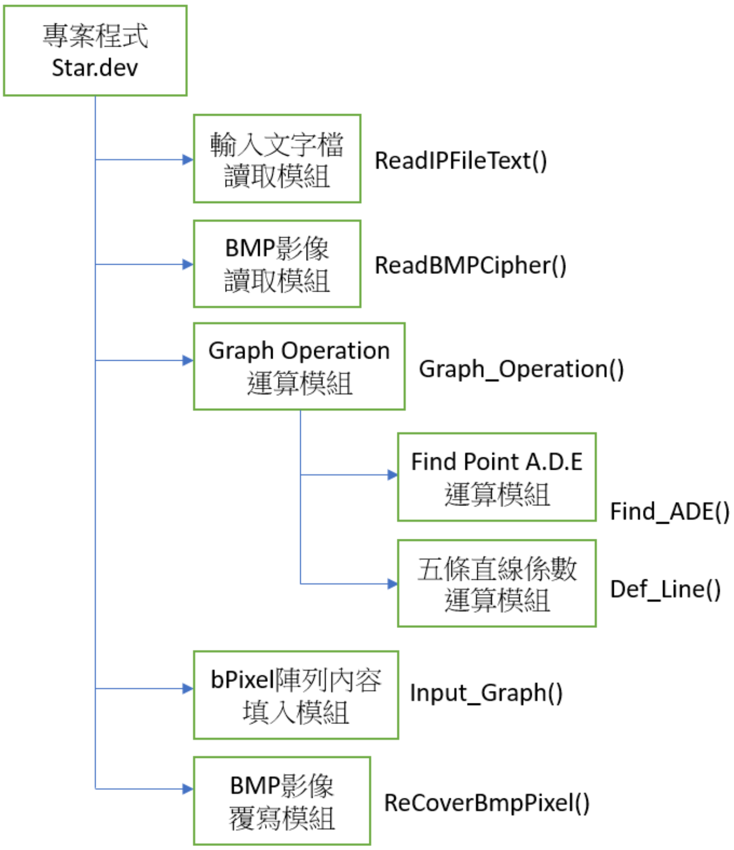


- 輸出文字檔三 Draw_Star3.bmp

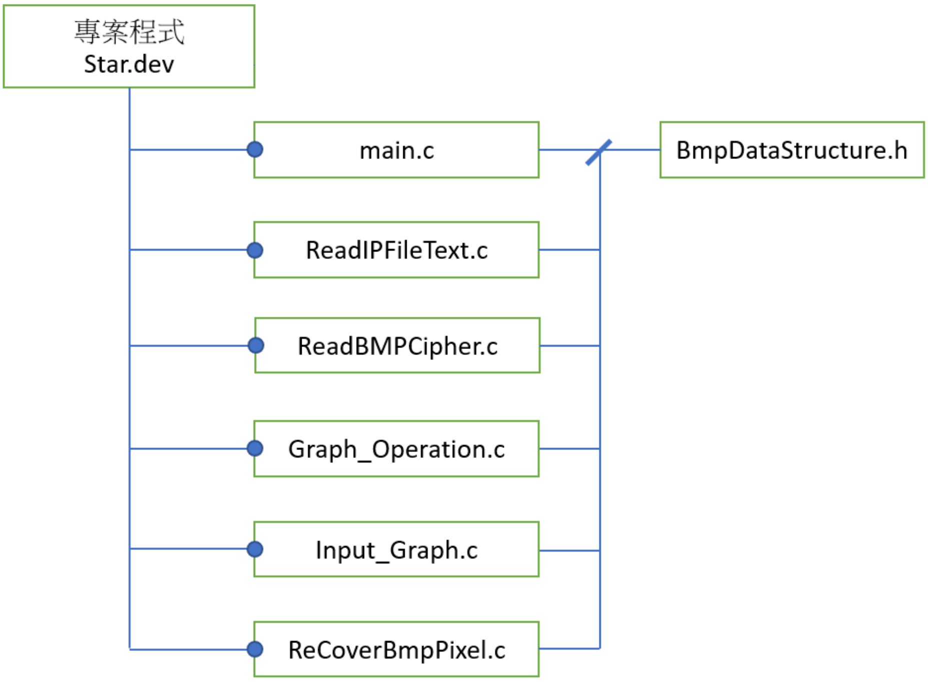


五、其他設計文件

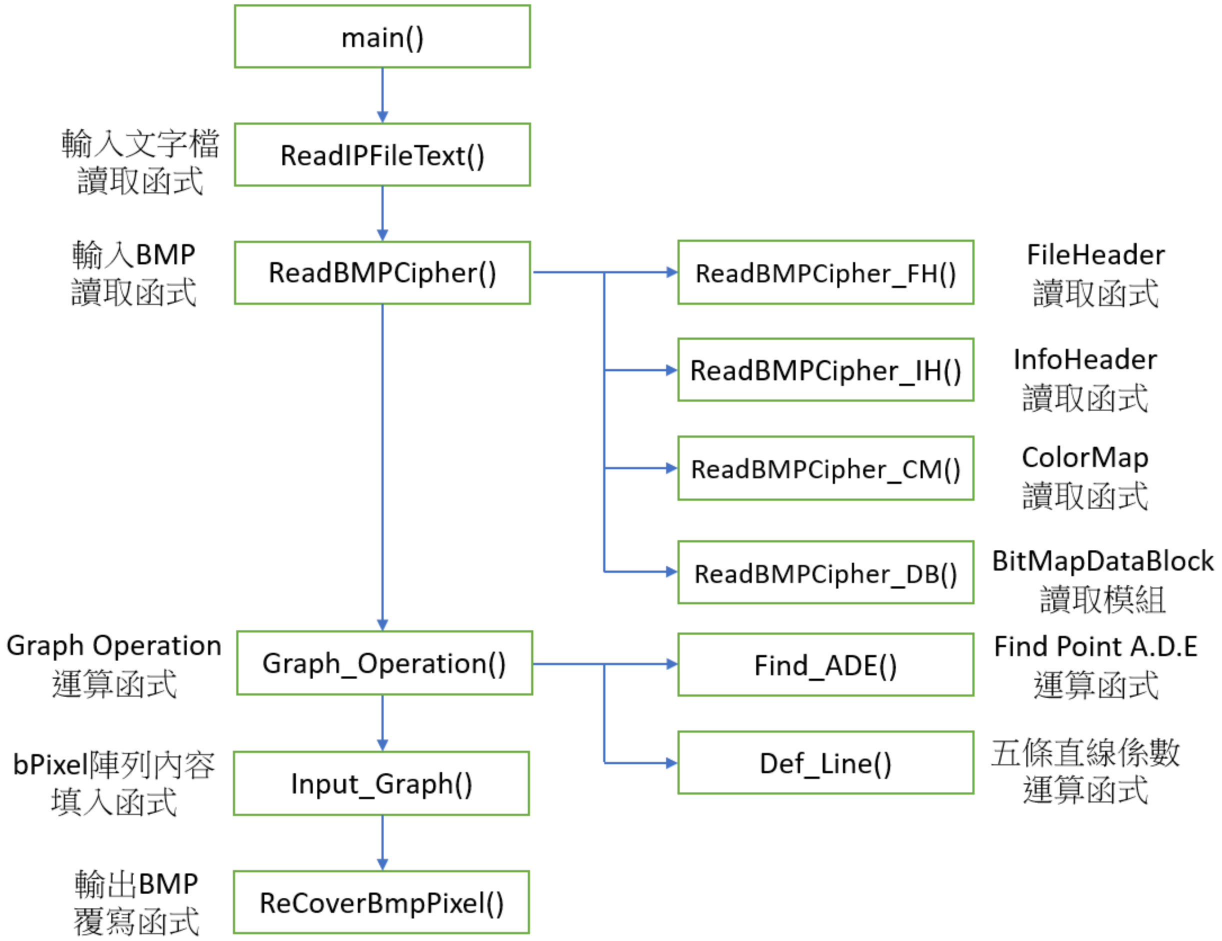
1. 專案程式模組架構圖



2. 專案檔案關係圖

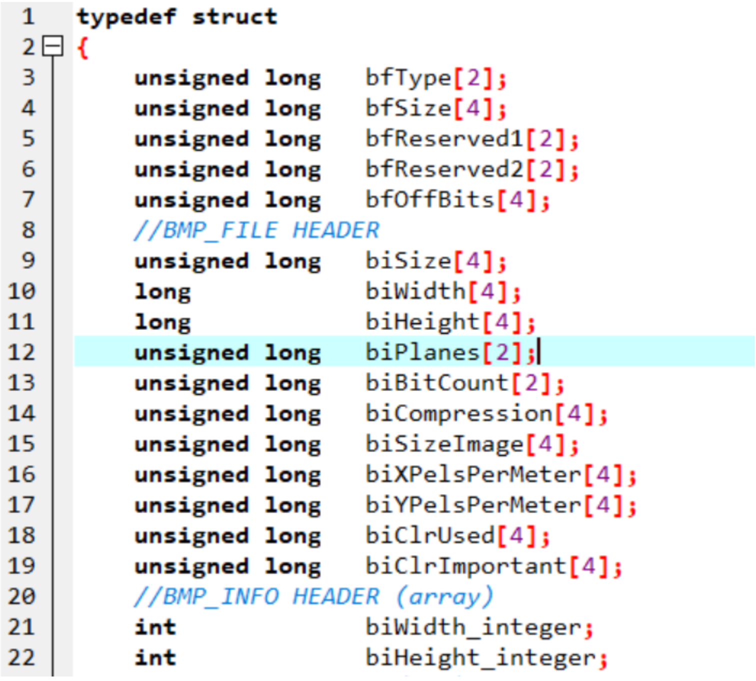


3.　函式呼叫圖

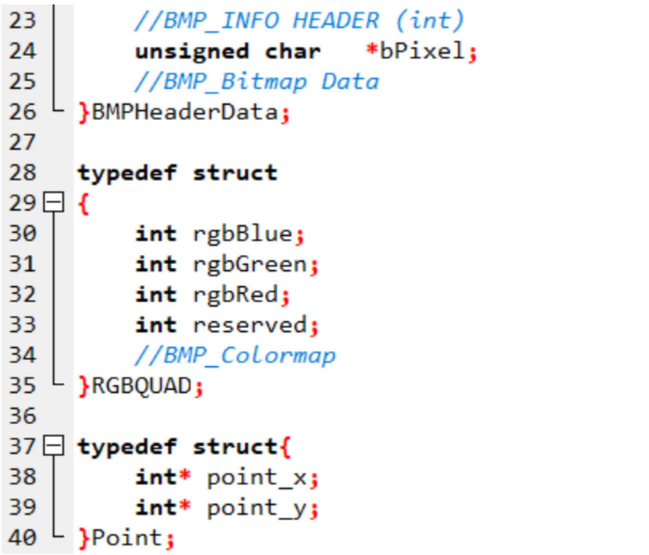

main()
輸入文字檔
讀取函式
ReadIPFileText()
輸入BMP
讀取函式
ReadBMPCipher()
ReadBMPCipher_FH()
FileHeader
讀取函式
ReadBMPCipher_IH()
InfoHeader
讀取函式
ReadBMPCipher_CM()
ColorMap
讀取函式
ReadBMPCipher_DB()
BitMapDataBlock
讀取模組
Graph Operation
運算函式
Graph_Operation()
Find_ADE()
Find Point A.D.E
運算函式
Def_Line()
五條直線係數
運算函式
bPixel陣列內容
填入函式
Input_Graph()
輸出BMP
覆寫函式
ReCoverBmpPixel()

六、附錄：原始程式列印

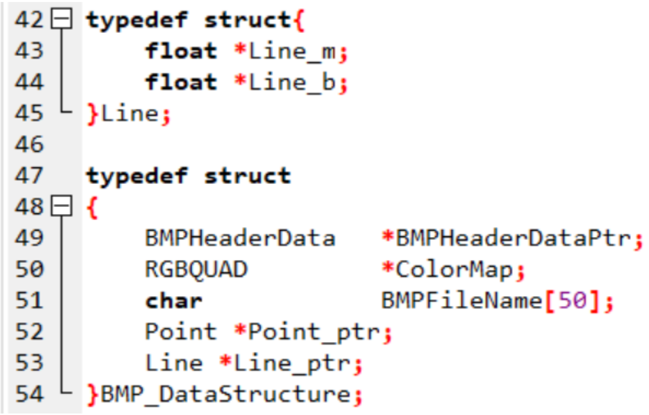
- BmpDataStructure.h



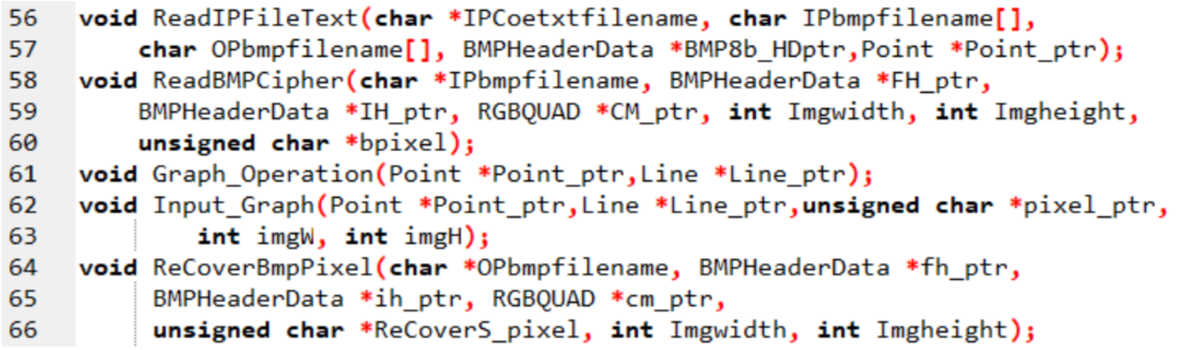
```c
typedef struct
{
    unsigned long   bfType[2];
    unsigned long   bfSize[4];
    unsigned long   bfReserved1[2];
    unsigned long   bfReserved2[2];
    unsigned long   bfOffBits[4];
    //BMP_FILE HEADER
    unsigned long   biSize[4];
    long            biWidth[4];
    long            biHeight[4];
    unsigned long   biPlanes[2];
    unsigned long   biBitCount[2];
    unsigned long   biCompression[4];
    unsigned long   biSizeImage[4];
    unsigned long   biXPelsPerMeter[4];
    unsigned long   biYPelsPerMeter[4];
    unsigned long   biClrUsed[4];
    unsigned long   biClrImportant[4];
    //BMP_INFO HEADER (array)
    int             biWidth_integer;
    int             biHeight_integer;
```



```c
    //BMP_INFO HEADER (int)
    unsigned char   *bPixel;
    //BMP_Bitmap Data
}BMPHeaderData;

typedef struct
{
    int rgbBlue;
    int rgbGreen;
    int rgbRed;
    int reserved;
    //BMP_Colormap
}RGBQUAD;

typedef struct{
    int* point_x;
    int* point_y;
}Point;
```

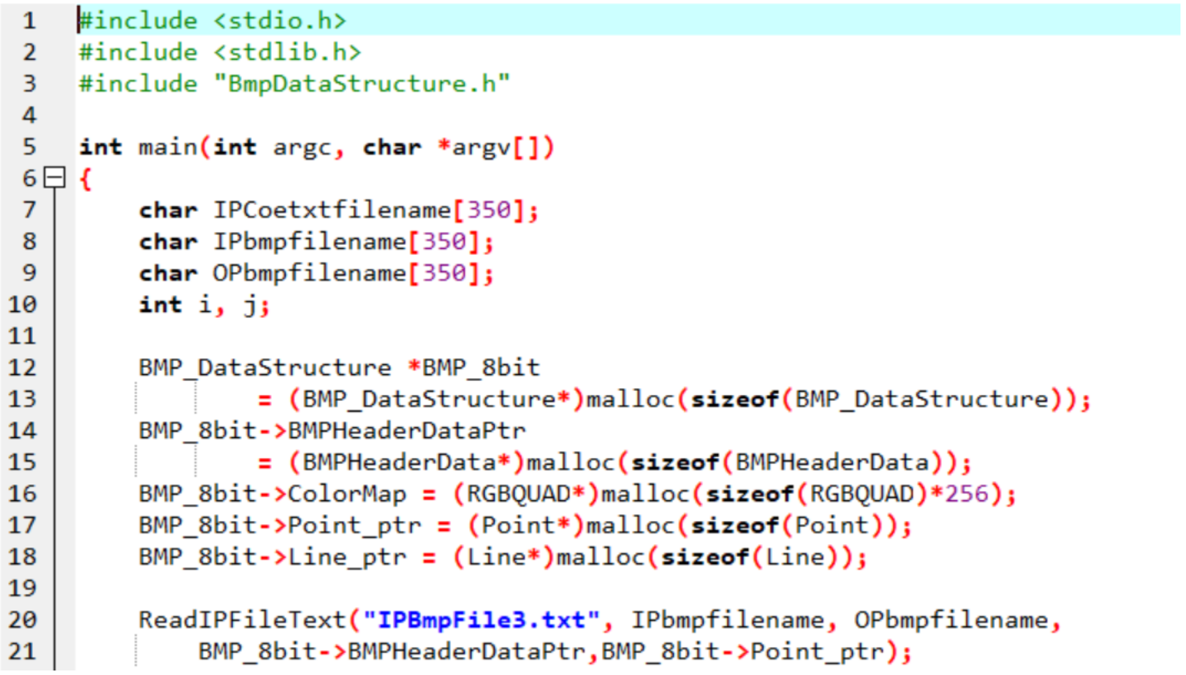


```c
typedef struct{
    float *Line_m;
    float *Line_b;
}Line;

typedef struct
{
    BMPHeaderData   *BMPHeaderDataPtr;
    RGBQUAD         *ColorMap;
    char            BMPFileName[50];
    Point *Point_ptr;
    Line *Line_ptr;
}BMP_DataStructure;
```

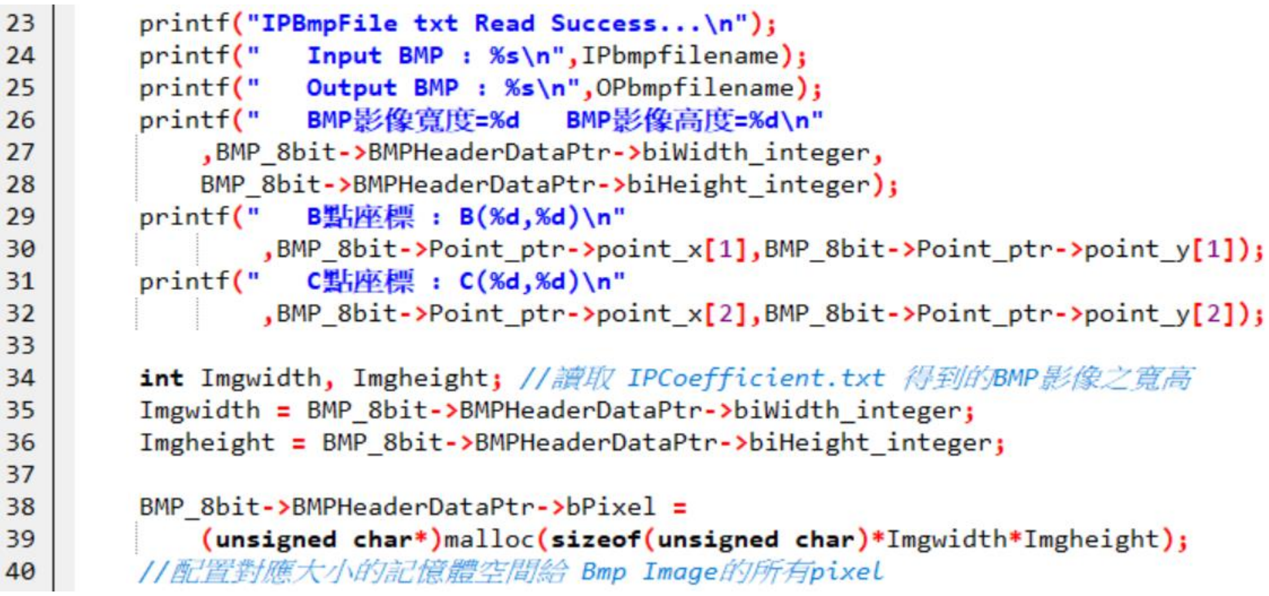


```c
void ReadIPFileText(char *IPCoetxtfilename, char IPbmpfilename[],
    char OPbmpfilename[], BMPHeaderData *BMP8b_HDptr,Point *Point_ptr);
void ReadBMPCipher(char *IPbmpfilename, BMPHeaderData *FH_ptr,
    BMPHeaderData *IH_ptr, RGBQUAD *CM_ptr, int Imgwidth, int Imgheight,
    unsigned char *bpixel);
void Graph_Operation(Point *Point_ptr,Line *Line_ptr);
void Input_Graph(Point *Point_ptr,Line *Line_ptr,unsigned char *pixel_ptr,
        int imgW, int imgH);
void ReCoverBmpPixel(char *OPbmpfilename, BMPHeaderData *fh_ptr,
     BMPHeaderData *ih_ptr, RGBQUAD *cm_ptr,
     unsigned char *ReCoverS_pixel, int Imgwidth, int Imgheight);
```

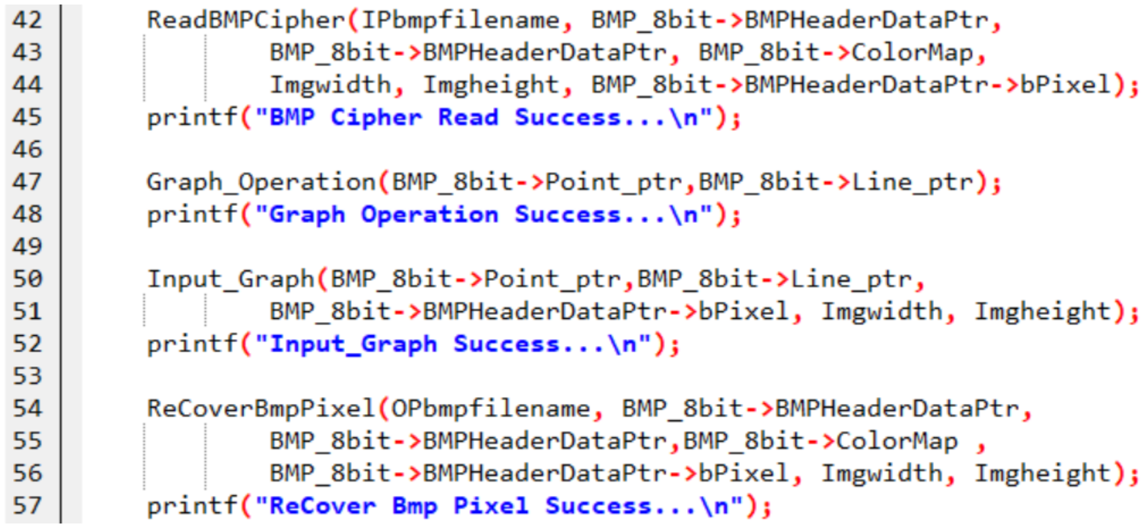
- main.c



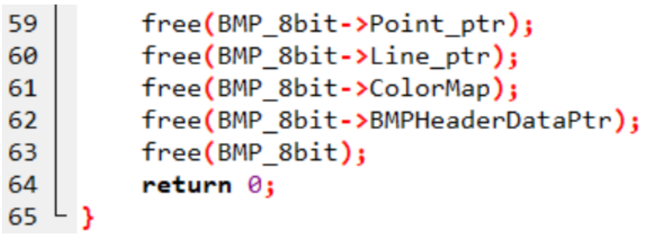
```c
#include <stdio.h>
#include <stdlib.h>
#include "BmpDataStructure.h"

int main(int argc, char *argv[])
{
    char IPCoetxtfilename[350];
    char IPbmpfilename[350];
    char OPbmpfilename[350];
    int i, j;

    BMP_DataStructure *BMP_8bit
            = (BMP_DataStructure*)malloc(sizeof(BMP_DataStructure));
    BMP_8bit->BMPHeaderDataPtr
            = (BMPHeaderData*)malloc(sizeof(BMPHeaderData));
    BMP_8bit->ColorMap = (RGBQUAD*)malloc(sizeof(RGBQUAD)*256);
    BMP_8bit->Point_ptr = (Point*)malloc(sizeof(Point));
    BMP_8bit->Line_ptr = (Line*)malloc(sizeof(Line));

    ReadIPFileText("IPBmpFile3.txt", IPbmpfilename, OPbmpfilename,
        BMP_8bit->BMPHeaderDataPtr,BMP_8bit->Point_ptr);
```



```c
    printf("IPBmpFile txt Read Success...\n");
    printf("   Input BMP : %s\n",IPbmpfilename);
    printf("   Output BMP : %s\n",OPbmpfilename);
    printf("   BMP影像寬度=%d   BMP影像高度=%d\n"
        ,BMP_8bit->BMPHeaderDataPtr->biWidth_integer,
        BMP_8bit->BMPHeaderDataPtr->biHeight_integer);
    printf("   B點座標 : B(%d,%d)\n"
            ,BMP_8bit->Point_ptr->point_x[1],BMP_8bit->Point_ptr->point_y[1]);
    printf("   C點座標 : C(%d,%d)\n"
            ,BMP_8bit->Point_ptr->point_x[2],BMP_8bit->Point_ptr->point_y[2]);

    int Imgwidth, Imgheight; //讀取 IPCoefficient.txt 得到的BMP影像之寬高
    Imgwidth = BMP_8bit->BMPHeaderDataPtr->biWidth_integer;
    Imgheight = BMP_8bit->BMPHeaderDataPtr->biHeight_integer;

    BMP_8bit->BMPHeaderDataPtr->bPixel =
        (unsigned char*)malloc(sizeof(unsigned char)*Imgwidth*Imgheight);
    //配置對應大小的記憶體空間給 Bmp Image的所有pixel
```

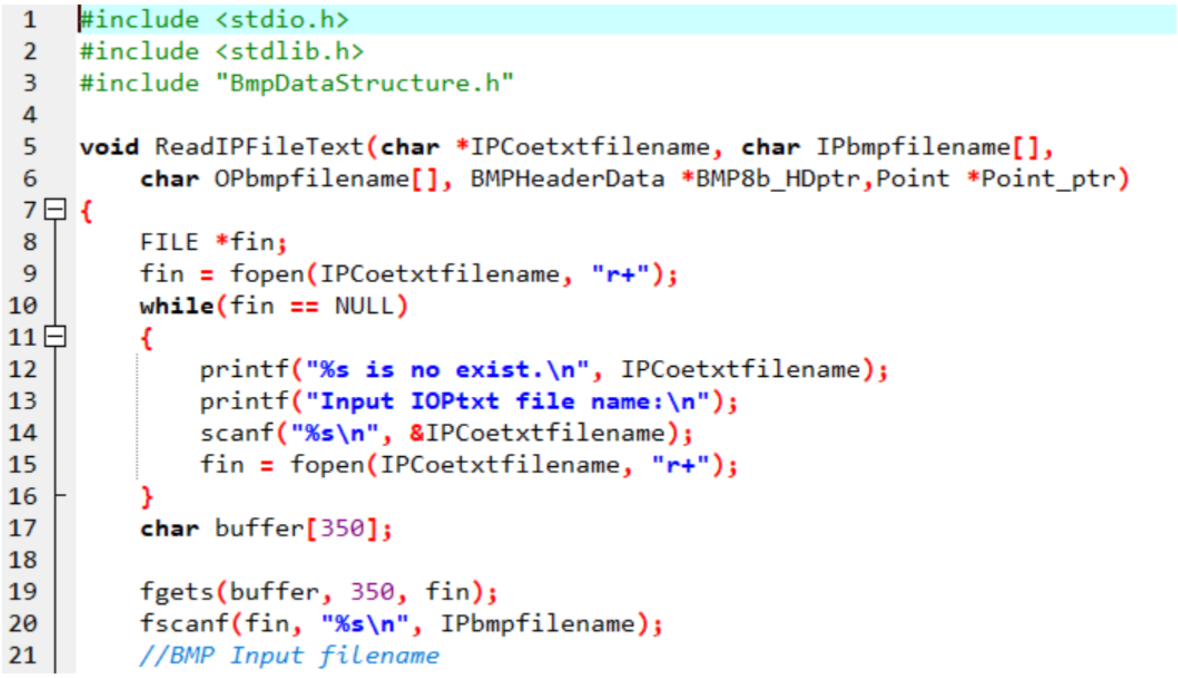


```c
    ReadBMPCipher(IPbmpfilename, BMP_8bit->BMPHeaderDataPtr,
            BMP_8bit->BMPHeaderDataPtr, BMP_8bit->ColorMap,
            Imgwidth, Imgheight, BMP_8bit->BMPHeaderDataPtr->bPixel);
    printf("BMP Cipher Read Success...\n");

    Graph_Operation(BMP_8bit->Point_ptr,BMP_8bit->Line_ptr);
    printf("Graph Operation Success...\n");

    Input_Graph(BMP_8bit->Point_ptr,BMP_8bit->Line_ptr,
            BMP_8bit->BMPHeaderDataPtr->bPixel, Imgwidth, Imgheight);
    printf("Input_Graph Success...\n");

    ReCoverBmpPixel(OPbmpfilename, BMP_8bit->BMPHeaderDataPtr,
            BMP_8bit->BMPHeaderDataPtr,BMP_8bit->ColorMap ,
            BMP_8bit->BMPHeaderDataPtr->bPixel, Imgwidth, Imgheight);
    printf("ReCover Bmp Pixel Success...\n");
```

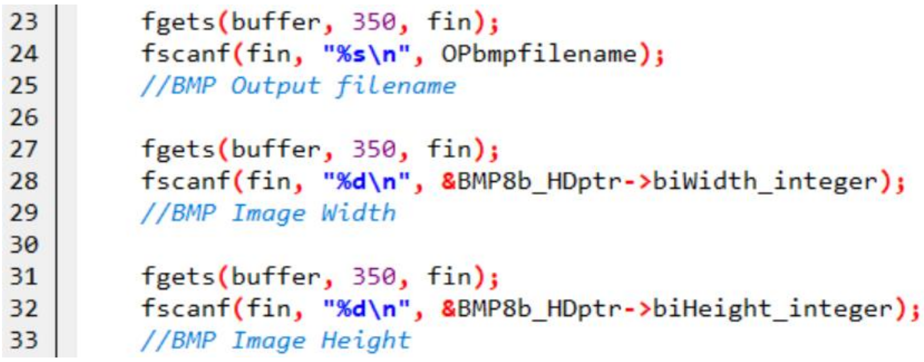


```c
    free(BMP_8bit->Point_ptr);
    free(BMP_8bit->Line_ptr);
    free(BMP_8bit->ColorMap);
    free(BMP_8bit->BMPHeaderDataPtr);
    free(BMP_8bit);
    return 0;
}
```

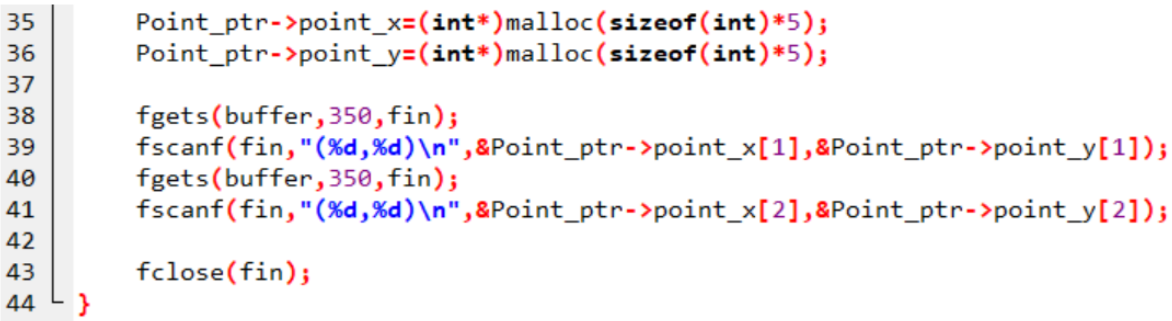
- ReadIPFileText.c



```c
#include <stdio.h>
#include <stdlib.h>
#include "BmpDataStructure.h"

void ReadIPFileText(char *IPCoetxtfilename, char IPbmpfilename[],
    char OPbmpfilename[], BMPHeaderData *BMP8b_HDptr,Point *Point_ptr)
{
    FILE *fin;
    fin = fopen(IPCoetxtfilename, "r+");
    while(fin == NULL)
    {
        printf("%s is no exist.\n", IPCoetxtfilename);
        printf("Input IOPtxt file name:\n");
        scanf("%s\n", &IPCoetxtfilename);
        fin = fopen(IPCoetxtfilename, "r+");
    }
    char buffer[350];

    fgets(buffer, 350, fin);
    fscanf(fin, "%s\n", IPbmpfilename);
    //BMP Input filename
```

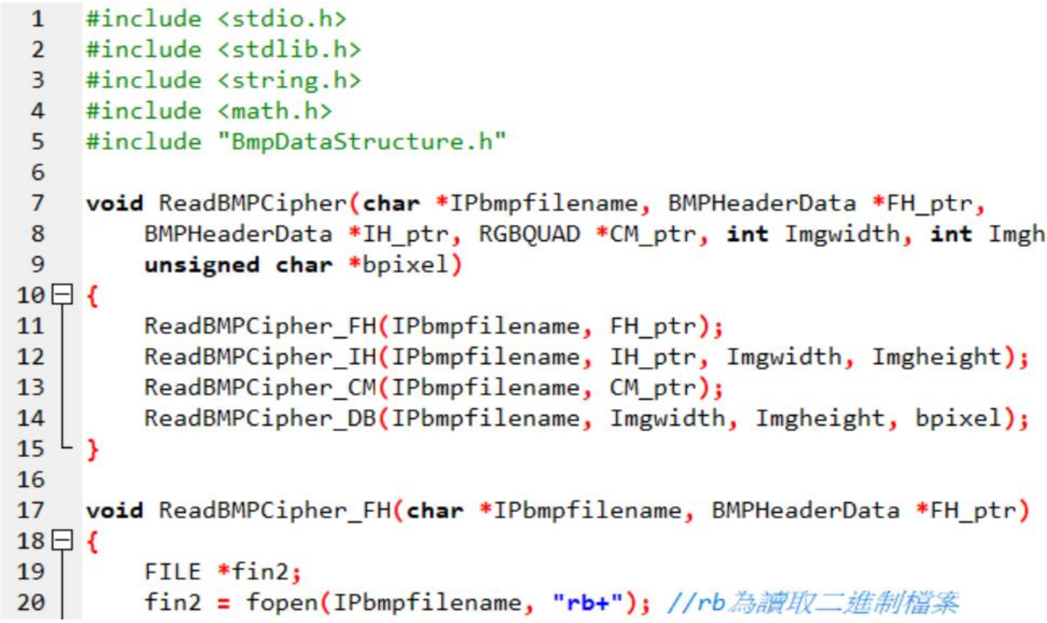


```c
    fgets(buffer, 350, fin);
    fscanf(fin, "%s\n", OPbmpfilename);
    //BMP Output filename

    fgets(buffer, 350, fin);
    fscanf(fin, "%d\n", &BMP8b_HDptr->biWidth_integer);
    //BMP Image Width

    fgets(buffer, 350, fin);
    fscanf(fin, "%d\n", &BMP8b_HDptr->biHeight_integer);
    //BMP Image Height
```

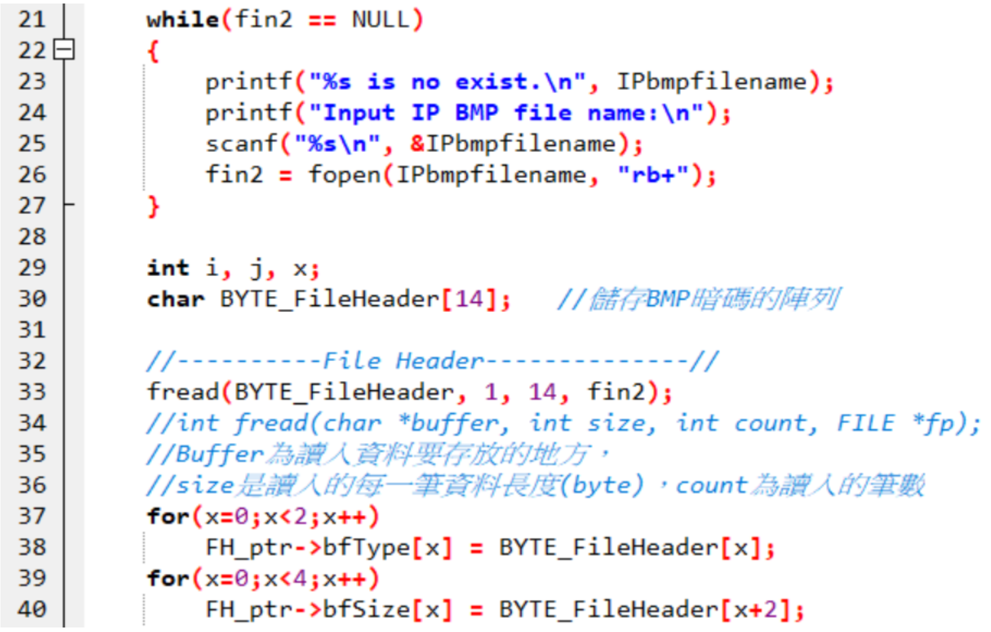


```c
    Point_ptr->point_x=(int*)malloc(sizeof(int)*5);
    Point_ptr->point_y=(int*)malloc(sizeof(int)*5);

    fgets(buffer,350,fin);
    fscanf(fin,"(%d,%d)\n",&Point_ptr->point_x[1],&Point_ptr->point_y[1]);
    fgets(buffer,350,fin);
    fscanf(fin,"(%d,%d)\n",&Point_ptr->point_x[2],&Point_ptr->point_y[2]);

    fclose(fin);
}
```

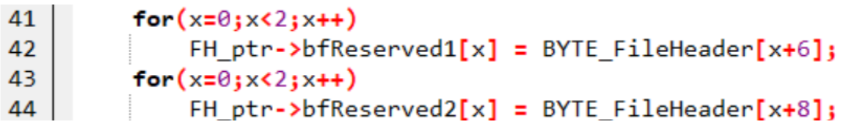
- ReadBMPCipher.c



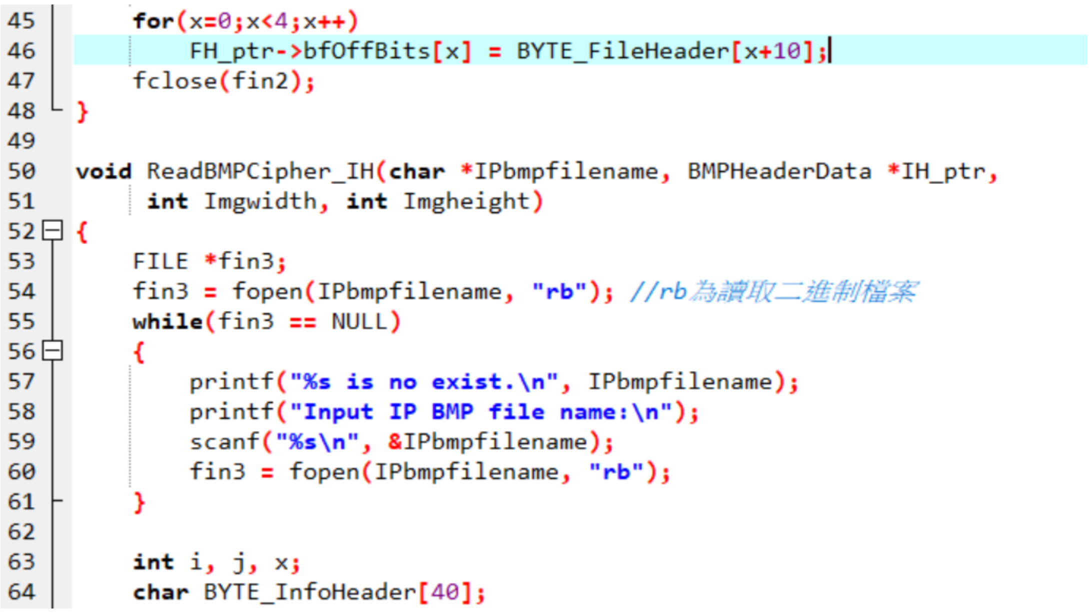
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "BmpDataStructure.h"

void ReadBMPCipher(char *IPbmpfilename, BMPHeaderData *FH_ptr,
    BMPHeaderData *IH_ptr, RGBQUAD *CM_ptr, int Imgwidth, int Imgheight,
    unsigned char *bpixel)
{
    ReadBMPCipher_FH(IPbmpfilename, FH_ptr);
    ReadBMPCipher_IH(IPbmpfilename, IH_ptr, Imgwidth, Imgheight);
    ReadBMPCipher_CM(IPbmpfilename, CM_ptr);
    ReadBMPCipher_DB(IPbmpfilename, Imgwidth, Imgheight, bpixel);
}

void ReadBMPCipher_FH(char *IPbmpfilename, BMPHeaderData *FH_ptr)
{
    FILE *fin2;
    fin2 = fopen(IPbmpfilename, "rb+"); //rb為讀取二進制檔案
```



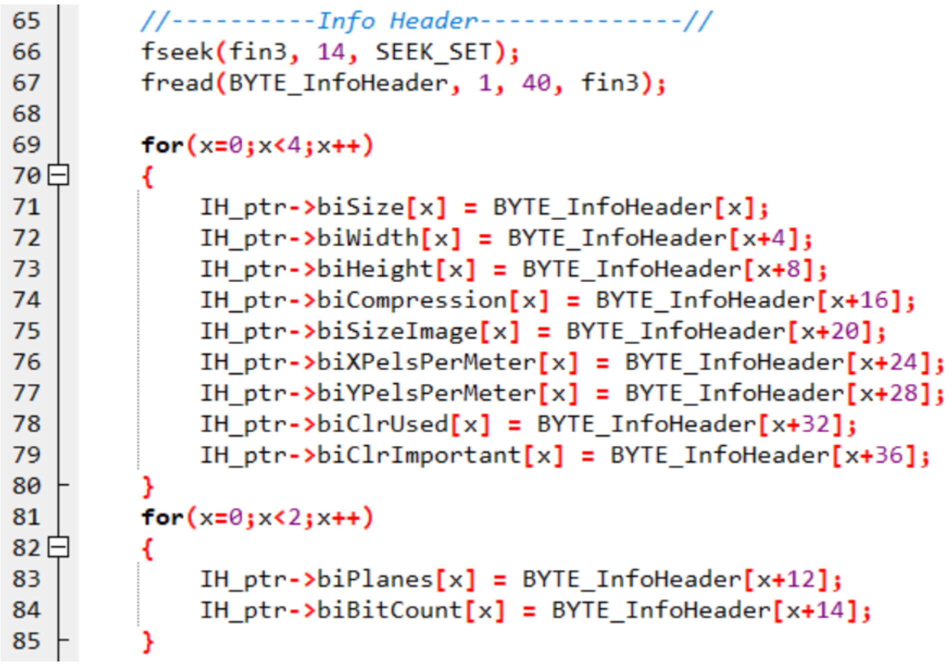
```c
    while(fin2 == NULL)
    {
        printf("%s is no exist.\n", IPbmpfilename);
        printf("Input IP BMP file name:\n");
        scanf("%s\n", &IPbmpfilename);
        fin2 = fopen(IPbmpfilename, "rb+");
    }

    int i, j, x;
    char BYTE_FileHeader[14];    //儲存BMP暗碼的陣列

    //----------File Header--------------//
    fread(BYTE_FileHeader, 1, 14, fin2);
    //int fread(char *buffer, int size, int count, FILE *fp);
    //Buffer為讀入資料要存放的地方，
    //size是讀入的每一筆資料長度(byte)，count為讀入的筆數
    for(x=0;x<2;x++)
        FH_ptr->bfType[x] = BYTE_FileHeader[x];
    for(x=0;x<4;x++)
        FH_ptr->bfSize[x] = BYTE_FileHeader[x+2];
```



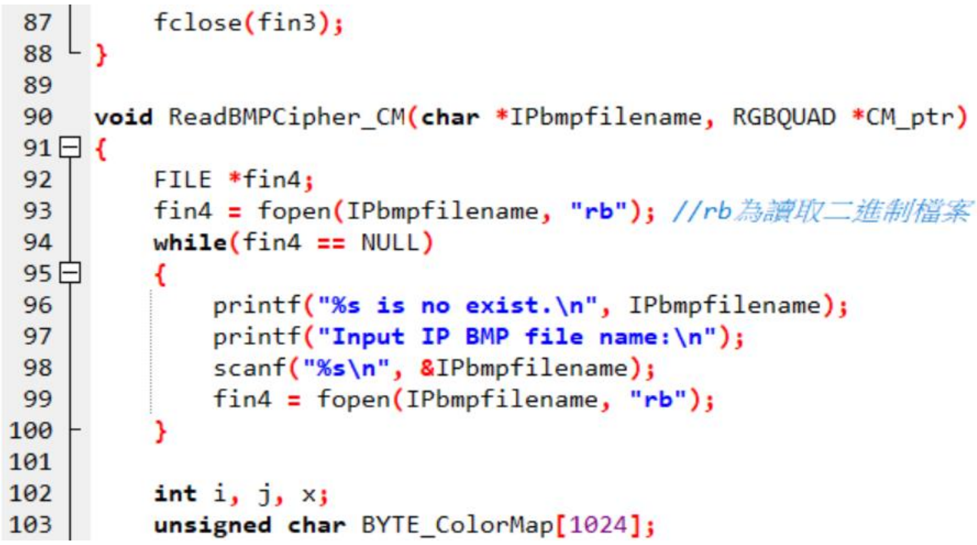
```c
    for(x=0;x<2;x++)
        FH_ptr->bfReserved1[x] = BYTE_FileHeader[x+6];
    for(x=0;x<2;x++)
        FH_ptr->bfReserved2[x] = BYTE_FileHeader[x+8];
```



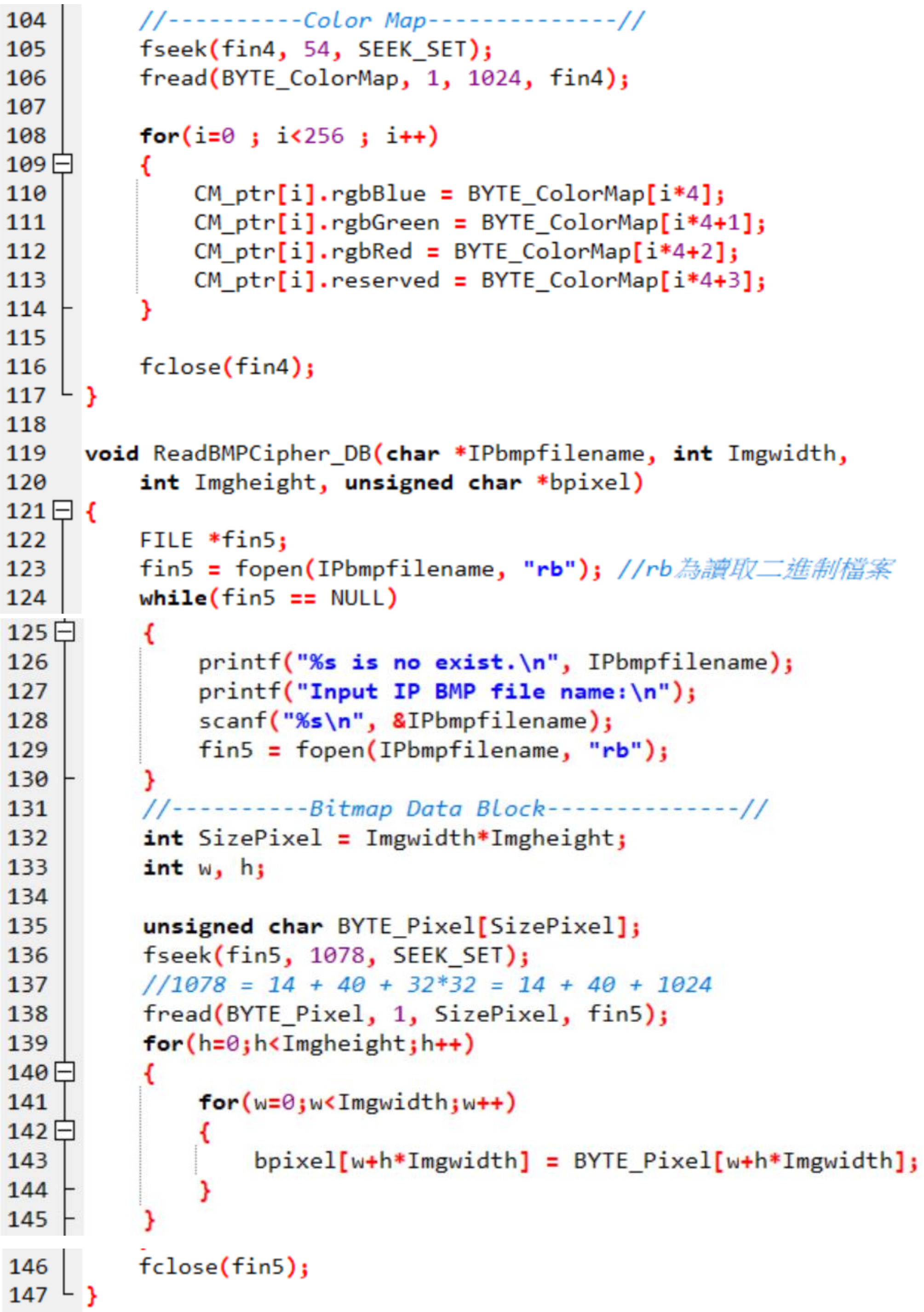
```c
    for(x=0;x<4;x++)
        FH_ptr->bfOffBits[x] = BYTE_FileHeader[x+10];
    fclose(fin2);
}

void ReadBMPCipher_IH(char *IPbmpfilename, BMPHeaderData *IH_ptr,
     int Imgwidth, int Imgheight)
{
    FILE *fin3;
    fin3 = fopen(IPbmpfilename, "rb"); //rb為讀取二進制檔案
    while(fin3 == NULL)
    {
        printf("%s is no exist.\n", IPbmpfilename);
        printf("Input IP BMP file name:\n");
        scanf("%s\n", &IPbmpfilename);
        fin3 = fopen(IPbmpfilename, "rb");
    }

    int i, j, x;
    char BYTE_InfoHeader[40];
```



```c
    //----------Info Header--------------//
    fseek(fin3, 14, SEEK_SET);
    fread(BYTE_InfoHeader, 1, 40, fin3);

    for(x=0;x<4;x++)
    {
        IH_ptr->biSize[x] = BYTE_InfoHeader[x];
        IH_ptr->biWidth[x] = BYTE_InfoHeader[x+4];
        IH_ptr->biHeight[x] = BYTE_InfoHeader[x+8];
        IH_ptr->biCompression[x] = BYTE_InfoHeader[x+16];
        IH_ptr->biSizeImage[x] = BYTE_InfoHeader[x+20];
        IH_ptr->biXPelsPerMeter[x] = BYTE_InfoHeader[x+24];
        IH_ptr->biYPelsPerMeter[x] = BYTE_InfoHeader[x+28];
        IH_ptr->biClrUsed[x] = BYTE_InfoHeader[x+32];
        IH_ptr->biClrImportant[x] = BYTE_InfoHeader[x+36];
    }
    for(x=0;x<2;x++)
    {
        IH_ptr->biPlanes[x] = BYTE_InfoHeader[x+12];
        IH_ptr->biBitCount[x] = BYTE_InfoHeader[x+14];
    }
```

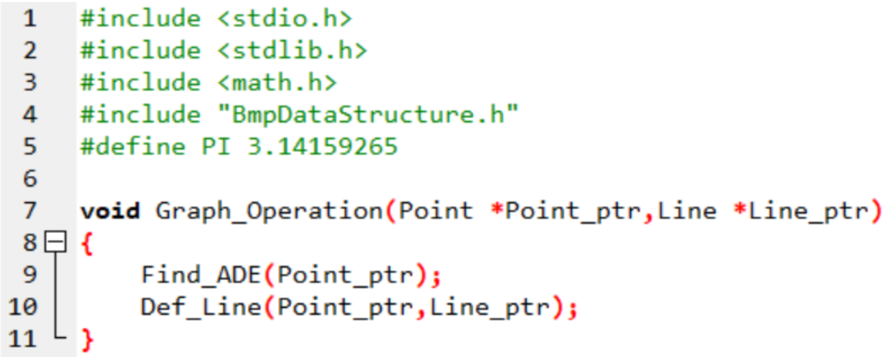


```c
    fclose(fin3);
}

void ReadBMPCipher_CM(char *IPbmpfilename, RGBQUAD *CM_ptr)
{
    FILE *fin4;
    fin4 = fopen(IPbmpfilename, "rb"); //rb為讀取二進制檔案
    while(fin4 == NULL)
    {
        printf("%s is no exist.\n", IPbmpfilename);
        printf("Input IP BMP file name:\n");
        scanf("%s\n", &IPbmpfilename);
        fin4 = fopen(IPbmpfilename, "rb");
    }

    int i, j, x;
    unsigned char BYTE_ColorMap[1024];
```

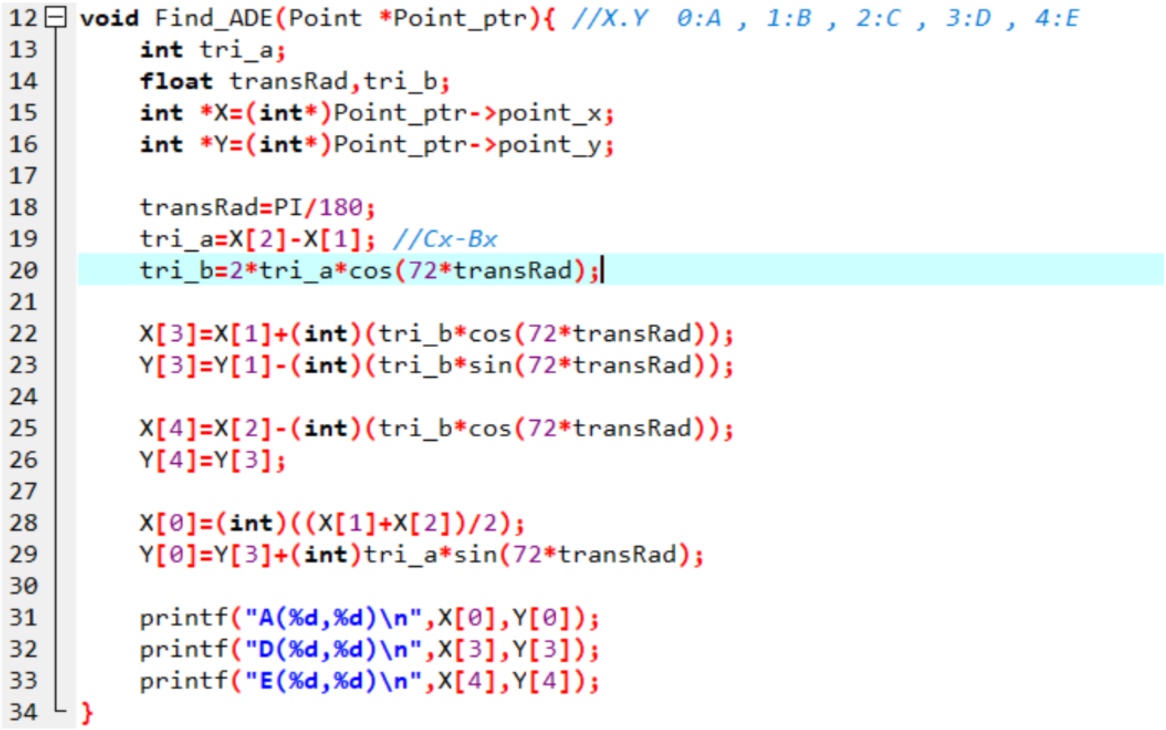


```c
    //----------Color Map-------------//
    fseek(fin4, 54, SEEK_SET);
    fread(BYTE_ColorMap, 1, 1024, fin4);

    for(i=0 ; i<256 ; i++)
    {
        CM_ptr[i].rgbBlue = BYTE_ColorMap[i*4];
        CM_ptr[i].rgbGreen = BYTE_ColorMap[i*4+1];
        CM_ptr[i].rgbRed = BYTE_ColorMap[i*4+2];
        CM_ptr[i].reserved = BYTE_ColorMap[i*4+3];
    }

    fclose(fin4);
}

void ReadBMPCipher_DB(char *IPbmpfilename, int Imgwidth,
    int Imgheight, unsigned char *bpixel)
{
    FILE *fin5;
    fin5 = fopen(IPbmpfilename, "rb"); //rb為讀取二進制檔案
    while(fin5 == NULL)
    {
        printf("%s is no exist.\n", IPbmpfilename);
        printf("Input IP BMP file name:\n");
        scanf("%s\n", &IPbmpfilename);
        fin5 = fopen(IPbmpfilename, "rb");
    }
    //----------Bitmap Data Block-------------//
    int SizePixel = Imgwidth*Imgheight;
    int w, h;

    unsigned char BYTE_Pixel[SizePixel];
    fseek(fin5, 1078, SEEK_SET);
    //1078 = 14 + 40 + 32*32 = 14 + 40 + 1024
    fread(BYTE_Pixel, 1, SizePixel, fin5);
    for(h=0;h<Imgheight;h++)
    {
        for(w=0;w<Imgwidth;w++)
        {
            bpixel[w+h*Imgwidth] = BYTE_Pixel[w+h*Imgwidth];
        }
    }
    fclose(fin5);
}
```

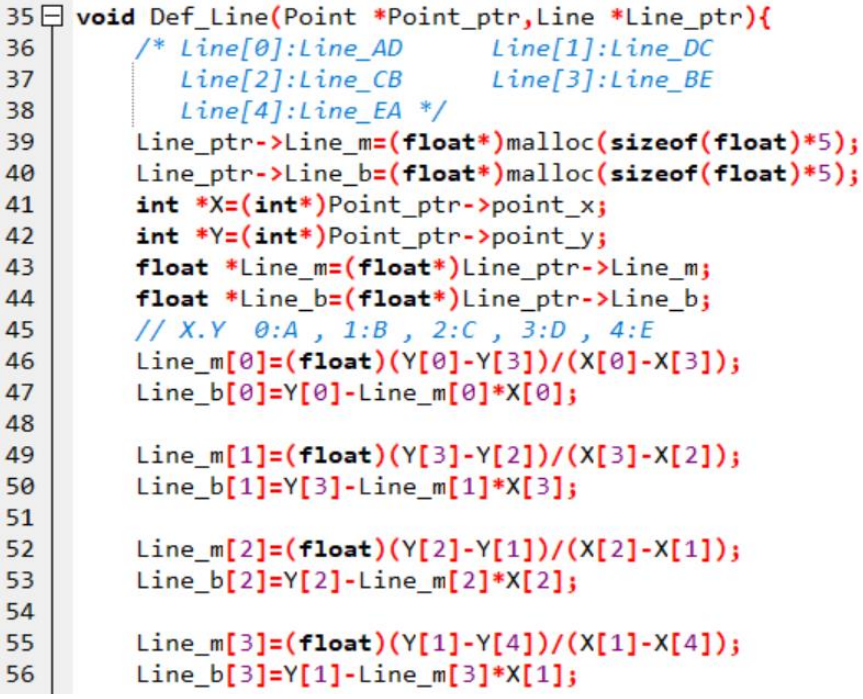
- Graph_Operation.c



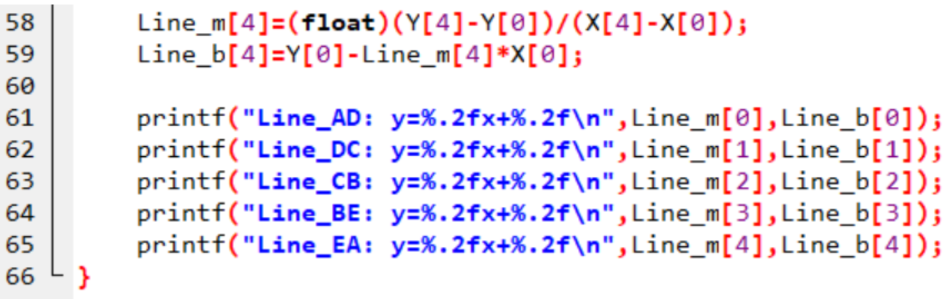
```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "BmpDataStructure.h"
#define PI 3.14159265

void Graph_Operation(Point *Point_ptr,Line *Line_ptr)
{
    Find_ADE(Point_ptr);
    Def_Line(Point_ptr,Line_ptr);
}
```



```c
void Find_ADE(Point *Point_ptr){ //X.Y  0:A , 1:B , 2:C , 3:D , 4:E
    int tri_a;
    float transRad,tri_b;
    int *X=(int*)Point_ptr->point_x;
    int *Y=(int*)Point_ptr->point_y;

    transRad=PI/180;
    tri_a=X[2]-X[1]; //Cx-Bx
    tri_b=2*tri_a*cos(72*transRad);

    X[3]=X[1]+(int)(tri_b*cos(72*transRad));
    Y[3]=Y[1]-(int)(tri_b*sin(72*transRad));

    X[4]=X[2]-(int)(tri_b*cos(72*transRad));
    Y[4]=Y[3];

    X[0]=(int)((X[1]+X[2])/2);
    Y[0]=Y[3]+(int)tri_a*sin(72*transRad);

    printf("A(%d,%d)\n",X[0],Y[0]);
    printf("D(%d,%d)\n",X[3],Y[3]);
    printf("E(%d,%d)\n",X[4],Y[4]);
}
```

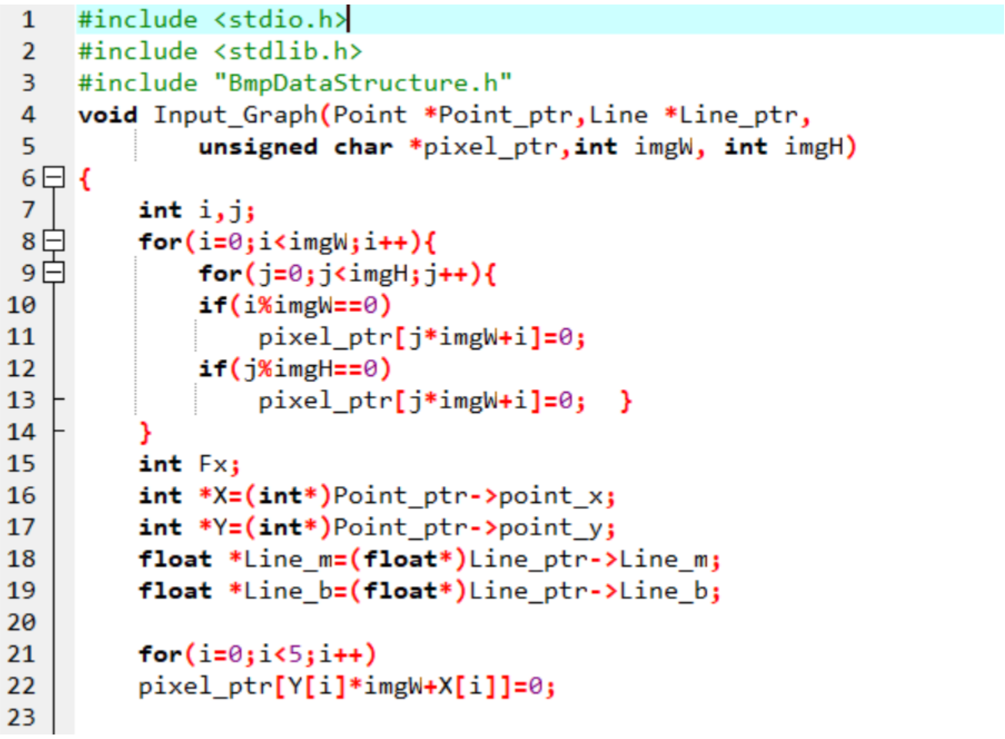


```c
void Def_Line(Point *Point_ptr,Line *Line_ptr){
    /* Line[0]:Line_AD      Line[1]:Line_DC
       Line[2]:Line_CB      Line[3]:Line_BE
       Line[4]:Line_EA */
    Line_ptr->Line_m=(float*)malloc(sizeof(float)*5);
    Line_ptr->Line_b=(float*)malloc(sizeof(float)*5);
    int *X=(int*)Point_ptr->point_x;
    int *Y=(int*)Point_ptr->point_y;
    float *Line_m=(float*)Line_ptr->Line_m;
    float *Line_b=(float*)Line_ptr->Line_b;
    // X.Y  0:A , 1:B , 2:C , 3:D , 4:E
    Line_m[0]=(float)(Y[0]-Y[3])/(X[0]-X[3]);
    Line_b[0]=Y[0]-Line_m[0]*X[0];

    Line_m[1]=(float)(Y[3]-Y[2])/(X[3]-X[2]);
    Line_b[1]=Y[3]-Line_m[1]*X[3];

    Line_m[2]=(float)(Y[2]-Y[1])/(X[2]-X[1]);
    Line_b[2]=Y[2]-Line_m[2]*X[2];

    Line_m[3]=(float)(Y[1]-Y[4])/(X[1]-X[4]);
    Line_b[3]=Y[1]-Line_m[3]*X[1];
```

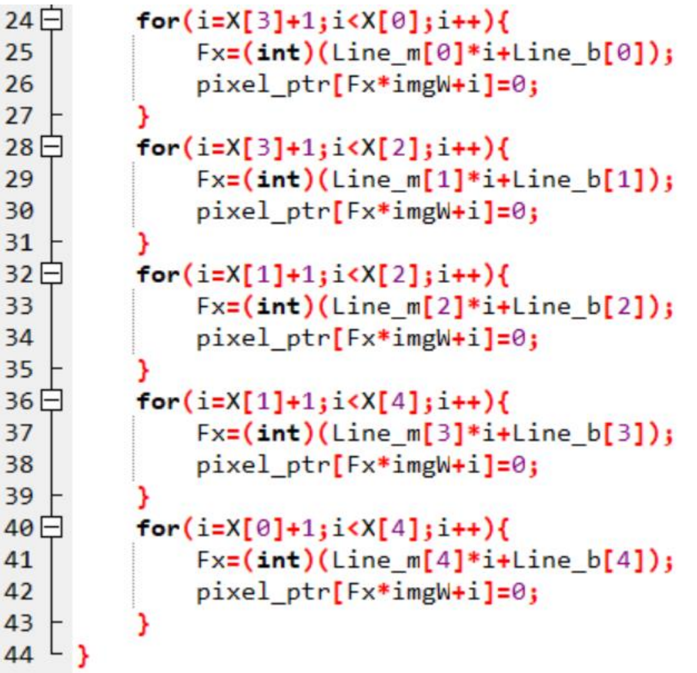


```c
    Line_m[4]=(float)(Y[4]-Y[0])/(X[4]-X[0]);
    Line_b[4]=Y[0]-Line_m[4]*X[0];

    printf("Line_AD: y=%.2fx+%.2f\n",Line_m[0],Line_b[0]);
    printf("Line_DC: y=%.2fx+%.2f\n",Line_m[1],Line_b[1]);
    printf("Line_CB: y=%.2fx+%.2f\n",Line_m[2],Line_b[2]);
    printf("Line_BE: y=%.2fx+%.2f\n",Line_m[3],Line_b[3]);
    printf("Line_EA: y=%.2fx+%.2f\n",Line_m[4],Line_b[4]);
}
```

- Input_Graph.c

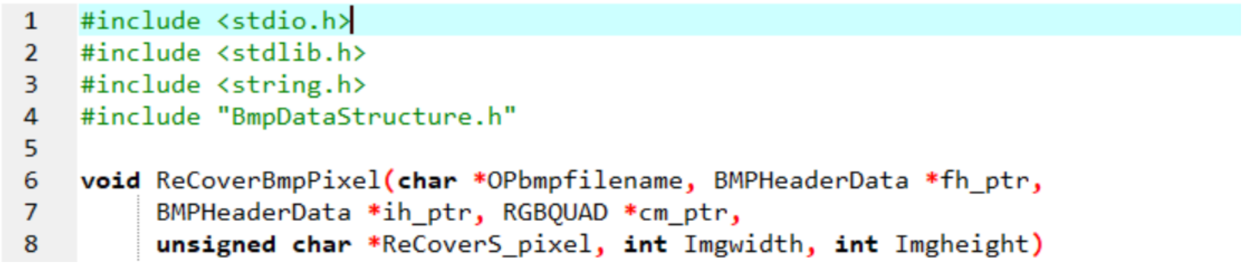


```c
#include <stdio.h>
#include <stdlib.h>
#include "BmpDataStructure.h"
void Input_Graph(Point *Point_ptr,Line *Line_ptr,
        unsigned char *pixel_ptr,int imgW, int imgH)
{
    int i,j;
    for(i=0;i<imgW;i++){
        for(j=0;j<imgH;j++){
        if(i%imgW==0)
            pixel_ptr[j*imgW+i]=0;
        if(j%imgH==0)
            pixel_ptr[j*imgW+i]=0;   }
    }
    int Fx;
    int *X=(int*)Point_ptr->point_x;
    int *Y=(int*)Point_ptr->point_y;
    float *Line_m=(float*)Line_ptr->Line_m;
    float *Line_b=(float*)Line_ptr->Line_b;

    for(i=0;i<5;i++)
    pixel_ptr[Y[i]*imgW+X[i]]=0;

```

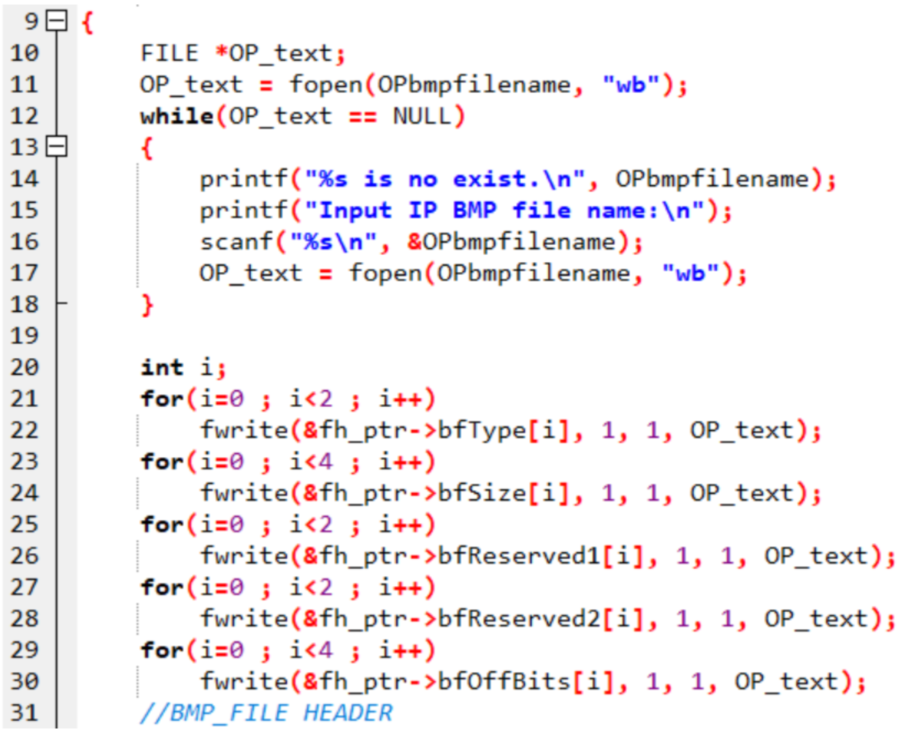


```c
    for(i=X[3]+1;i<X[0];i++){
        Fx=(int)(Line_m[0]*i+Line_b[0]);
        pixel_ptr[Fx*imgW+i]=0;
    }
    for(i=X[3]+1;i<X[2];i++){
        Fx=(int)(Line_m[1]*i+Line_b[1]);
        pixel_ptr[Fx*imgW+i]=0;
    }
    for(i=X[1]+1;i<X[2];i++){
        Fx=(int)(Line_m[2]*i+Line_b[2]);
        pixel_ptr[Fx*imgW+i]=0;
    }
    for(i=X[1]+1;i<X[4];i++){
        Fx=(int)(Line_m[3]*i+Line_b[3]);
        pixel_ptr[Fx*imgW+i]=0;
    }
    for(i=X[0]+1;i<X[4];i++){
        Fx=(int)(Line_m[4]*i+Line_b[4]);
        pixel_ptr[Fx*imgW+i]=0;
    }
}
```

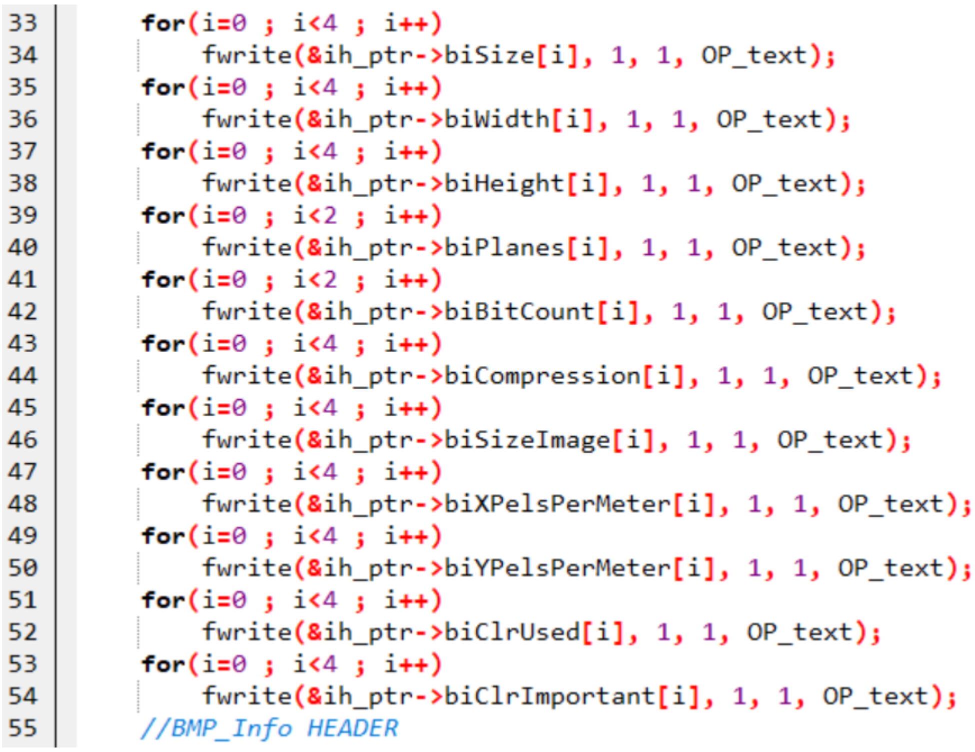
- ReCoverBmpPixel.c



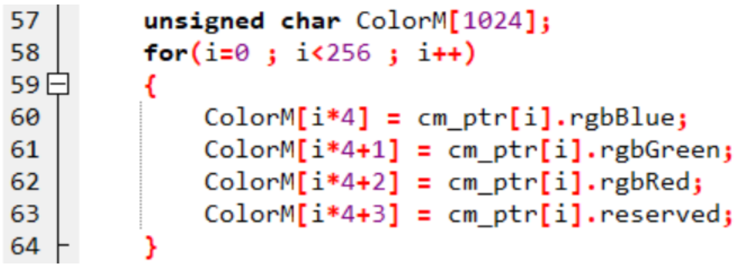
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "BmpDataStructure.h"

void ReCoverBmpPixel(char *OPbmpfilename, BMPHeaderData *fh_ptr,
     BMPHeaderData *ih_ptr, RGBQUAD *cm_ptr,
     unsigned char *ReCoverS_pixel, int Imgwidth, int Imgheight)
```



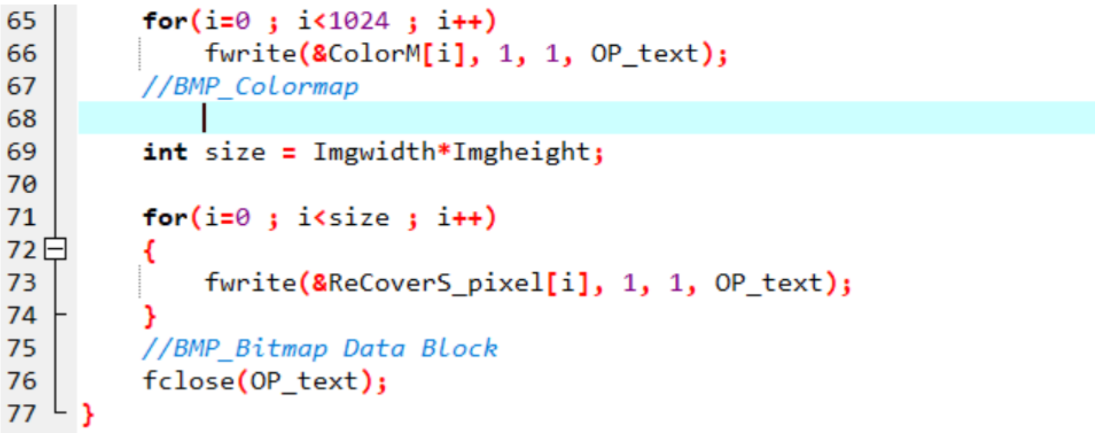
```
{
    FILE *OP_text;
    OP_text = fopen(OPbmpfilename, "wb");
    while(OP_text == NULL)
    {
        printf("%s is no exist.\n", OPbmpfilename);
        printf("Input IP BMP file name:\n");
        scanf("%s\n", &OPbmpfilename);
        OP_text = fopen(OPbmpfilename, "wb");
    }

    int i;
    for(i=0 ; i<2 ; i++)
        fwrite(&fh_ptr->bfType[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&fh_ptr->bfSize[i], 1, 1, OP_text);
    for(i=0 ; i<2 ; i++)
        fwrite(&fh_ptr->bfReserved1[i], 1, 1, OP_text);
    for(i=0 ; i<2 ; i++)
        fwrite(&fh_ptr->bfReserved2[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&fh_ptr->bfOffBits[i], 1, 1, OP_text);
    //BMP_FILE HEADER
```



```
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biSize[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biWidth[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biHeight[i], 1, 1, OP_text);
    for(i=0 ; i<2 ; i++)
        fwrite(&ih_ptr->biPlanes[i], 1, 1, OP_text);
    for(i=0 ; i<2 ; i++)
        fwrite(&ih_ptr->biBitCount[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biCompression[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biSizeImage[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biXPelsPerMeter[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biYPelsPerMeter[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biClrUsed[i], 1, 1, OP_text);
    for(i=0 ; i<4 ; i++)
        fwrite(&ih_ptr->biClrImportant[i], 1, 1, OP_text);
    //BMP_Info HEADER
```



```
    unsigned char ColorM[1024];
    for(i=0 ; i<256 ; i++)
    {
        ColorM[i*4] = cm_ptr[i].rgbBlue;
        ColorM[i*4+1] = cm_ptr[i].rgbGreen;
        ColorM[i*4+2] = cm_ptr[i].rgbRed;
        ColorM[i*4+3] = cm_ptr[i].reserved;
    }
```



```
    for(i=0 ; i<1024 ; i++)
        fwrite(&ColorM[i], 1, 1, OP_text);
    //BMP_Colormap

    int size = Imgwidth*Imgheight;

    for(i=0 ; i<size ; i++)
    {
        fwrite(&ReCoverS_pixel[i], 1, 1, OP_text);
    }
    //BMP_Bitmap Data Block
    fclose(OP_text);
}
```

七、心得與討論

本次程式所使用的副函式與嵌入式程式設計課程相同，兩者相比難度的差別在於，如何將該副函式結合 BMP 的檔案架構，合成一個完整的專案程式，另外兩者儲存的內容所使用的資料型態也不相同，須將這些差異一一克服，方可完成本次期末報告。