实习一 BMP 灰度图像压缩

一、实习目的与要求

【问题描述】

灰度图像的像素值范围在[0，255]之间，如果采用一个像素一个字节的存储方式，势必会造成空间的浪费。如果采用一定的无损压缩算法，可以大大提高减小文件大小，减少存储空间。本课题要求针对提供的 256 色（8 位）位图数据，采用教材上第 15 章动态规划中图像压缩算法（图像分段合并的思想），设计一个类，实现灰度位图数据的压缩和解压过程。

【基本要求】

一个完整的灰度图像类应具有以下功能：

（1）对 8 位位图数据的读功能，提供 ReadBitmap 方法。

ReadBitmap 方法有一个参数为输入位图文件名(*.bmp)，它能解析 8 位位图文件格式，获取位图 BITMAPINFOHEADER 信息和每个像素的数据信息，放入内存中。

（2）对 8 位位图数据的写功能，提供 WriteBitmap 方法。

WriteBitmap 方法有一个参数为输出位图文件名(*.bmp)，它能将内存中的位图文件信息，按照位图格式，写到位图文件中保存。

（3）灰度图像压缩功能，提供 Compress 方法。

Compress 方法有一个参数为输出压缩文件名(*.img) ，它能将已经装入到内存中的 8 位位图信息，进行压缩，形成段标题和以变长格式存储的像素的二进制串，写入到文件中（注意：Img 文件格式自行定义）。

（4）灰度图像解压功能，提供 UnCompress 方法。

UnCompress 方法有一个参数为输入压缩文件名(*.img)，它能解析 Img 文件格式，将其在内存中解压缩为 8 位位图信息，以便输出为位图文件。

（5）以上是该灰度图像类基本的四个方法，在实现时可根据需要扩充其他方法。在设计时，要使用面向对象的思想，考虑各个成员的访问权限。

【提高要求】

（1）基于 Windows 对话框界面，可选择输入/输出文件名，有压缩进度条显示。

（2）采用不同的数据集，比较其压缩比，采用最有效的压缩方式。

【测试数据】

lena.bmp，512*512*8

【测试用例】

类的测试用例如下：

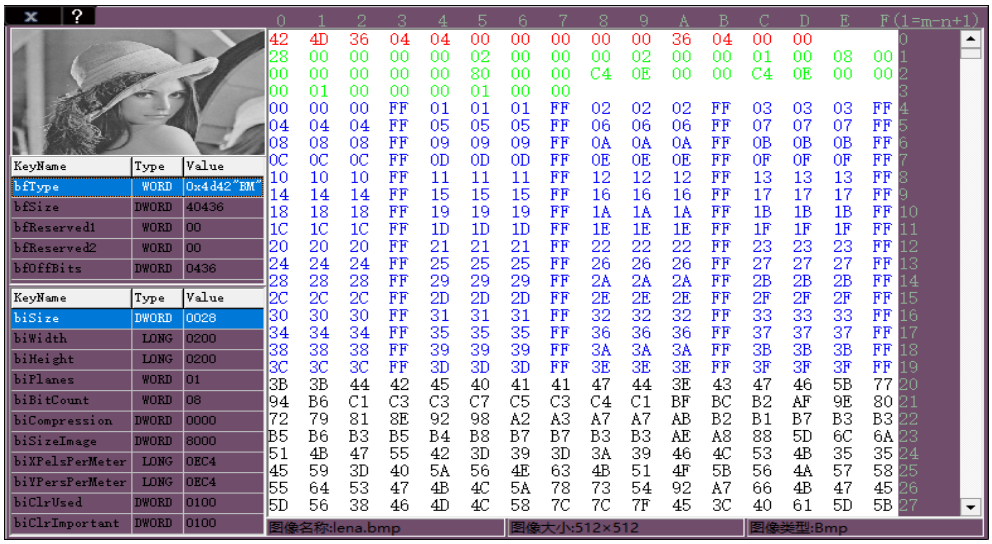
```
CCompressImage    Test;
Test. ReadBitmap(“数字化.bmp”);        读原始位图
Test. Compress(“Out.img”);              压缩
Test. UnCompress(“Out.img”);            解压
Test. WriteBitmap(“Out.bmp”);           还原位图信息
```

【测试结果】

可以使用 MD5 比较解压后的图与原图是否一样，验证你所实现的灰度图像类是否做到了无损压缩。

【实现提示】

有关 8 位的位图格式可以参考 MSDN 中 BITMAPINFOHEADER 结构的说明文档，注意其中 biBitCount=8 的说明。



变量名	地址偏移	大小	作用
bfType	0000h	2 bytes	说明文件的类型，可取值为： • BM – Windows 3.1x, 95, NT, ... • BA – OS/2 Bitmap Array • CI – OS/2 Color Icon • CP – OS/2 Color Pointer • IC – OS/2 Icon • PT – OS/2 Pointer
bfSize	0002h	4 bytes	说明该位图文件的大小，用字节为单位
bfReserved1	0006h	2 bytes	保留，必须设置为0
bfReserved2	0008h	2 bytes	保留，必须设置为0
bfOffBits	000Ah	4 bytes	说明从文件头开始到实际的图象数据之间的字节的偏移量。 这个参数是非常有用的，因为位图信息头和调色板的长度会根据不同情况而变化， 所以我们可以用这个偏移值迅速的从文件中读取到位图数据。

变量名	地址偏移	大小	作用
biSize	000Eh	4 bytes	BITMAPINFOHEADER结构所需要的字数。
biWidth	0012h	4 bytes	说明图像的宽度，用像素为单位
biHeight	0016h	4 bytes	说明图像的高度，以像素为单位。 注：这个值除了用于描述图像的高度之外，它还有另一个用处，就是指明该图像是倒内的位图，还是正向的位图。 如果该值是一个正数，说明图像是倒向的，如果该值是一个负数，则说明图像是正向的。 大多数的BMP文件都是倒向的位图，也就是高度值是一个正数。
biPlanes	001Ah	2 bytes	为目标设备说明颜色平面数， 其值将总是被设为1。
biBitCount	001Ch	2 bytes	说明比特数/像素，其值为1、4、8、16、24或32。
biCompression	001Eh	4 bytes	说明图像数据压缩的类型。取值范围： 0 BI_RGB 不压缩（最常用） 1 BI_RLE8 8比特游程编码（RLE），只用于8位位图 2 BI_RLE4 4比特游程编码（RLE），只用于4位位图 3 BI_BITFIELDS 比特域，用于16/32位位图 4 BI_JPEG JPEG 位图含JPEG图像（仅用于打印机） 5 BI_PNG PNG 位图含PNG图像（仅用于打印机）
biSizeImage	0022h	4 bytes	说明图像的大小， 以字节为单位。当用BI_RGB格式时，可设置为0。
biXPelsPerMeter	0026h	4 bytes	说明水平分辨率，用像素/米表示，有符号整数
biYPelsPerMeter	002Ah	4 bytes	说明垂直分辨率，用像素/米表示，有符号整数
biClrUsed	002Eh	4 bytes	说明位图实际使用的彩色表中的颜色索引数 （设为0的话，则说明使用所有调色板项）
biClrImportant	0032h	4 bytes	说明对图像显示有重要影响的颜色索引的数目 如果是0，表示都重要。

二、分析与设计

1、 需求分析与类设计

分析可以知道我们此题需要写两个主要的部分，一部分是操作，一部分是界面的显示，所以我设计了三个模块。第一个模块 BMPHead, 有两个类分别为 BITMAPFILEHEADER、BITMAPINFOHEADER；第二个模块 BMPCompress 也有两个类，分别为 CompressBmp、unCompressBmp，第三个模块 BMPWidget，有一个类 BmpWidget。下面是对它们的介绍

BITMAPFILEHEADER：负责储存文件头无太大用处

BITMAPINFOHEADER：负责储存文件信息头无太大用处

CompressBmp：
主要功能:
对 BMP 文件进行压缩
主要的方法：
1. readBMP() 负责读 BMP 文件
2. Compress() 对读取的文件进行 dp 操作，找出最优的储存位数
3. writeFile() 对 dp 操作后的数据进行写操作

unCompressBmp：
主要功能:
对 BMP 压缩后的文件进行解压
主要的方法：
1. unCompress() 负责读 BMP 压缩文件，并且进行解压

BmpWidget：主要负责显示界面以及读取文件名

2、 算法设计与分析

这题主要的核心算法是动态规划、写压缩文件的算法所以下文主要对这两个算法进行分析

变位压缩：
递推公式：s[i]=min{s[i-k]+k*bmax(i-k+1,i)}+11
bmax(i,j)=p[i]~p[j]中数字位数的最大值

写文件的算法：

在开始读文件的时候，就将 BMP 文件的头部信息写进文件去了，然后在 dp 完成后将分段数存储进去（分段数代表最终分成多少段），然后分段按照下列过程进行

1.写前面 11 位的段头分别位 b、l，当前位数(pos)如果大于 8，就写一次文件

```
L=self.c[i]
B=max(self.b[index+1:index+L+1])
buffer=(buffer<<3)^(B-1)
buffer=(buffer<<8)^(L-1)
pos+=11
```

2.开始读这段的数据，过程类似上面读一个数据判断当前位数，如果大于 8 位就进行写一次

```
buffer=(buffer<<B)^self.p[index]
```

3.最后要注意如果 pos 小于 8 位要在末尾补零

3、测试与改进

（1）功能测试

Lena.bmp 的压缩：

数字化.bmp 的压缩：

纹理.bmp 的压缩：

（2）性能测试

压缩：

1. Lena.bmp 压缩消耗的时间：28.69419503211975s
2. 数字化.bmp 压缩消耗的时间：31.14595866203308s
3.纹理.bmp 压缩消耗的时间：28.199455499649048s

解压：

1. Lena.zinp 压缩消耗的时间：0.22180414199829102s
2. 数字化.zinp 压缩消耗的时间：0.24996733665466309 s
3. 纹理.zinp 压缩消耗的时间：0.21869802474975586s

（3）改进与优化

1.读文件时，直接转化位了蛇形防止了加入中间数组的转化，节省空间。

2.解追踪时，采用了迭代而没有采用递归的方式，理论上可以防止递归太深。

3.将解压和压缩分离开，比将两个类写在一起更加好理解和操作。

三、实习小结

本次实习我们写了一个灰度 BMP 压缩的程序，在本次实习中我体会了动态规划的精妙的地方，并且又再一次体会到了文件读写带来的问题。我在此题中的收获主要是用 python 练了一下手，头一次用 python 写这样的完整的程序，加深了我对 python 的理解。对于本程序我觉得还可以有优化，虽然运算的速度并不能提高太多，但是可以加入批处理，如：一次处理多张 bmp 进行压缩，这样可以提高用户的体验，在以后有时间可以选择加入这样的功能。

成绩评定：

教师签名：

批改日期：