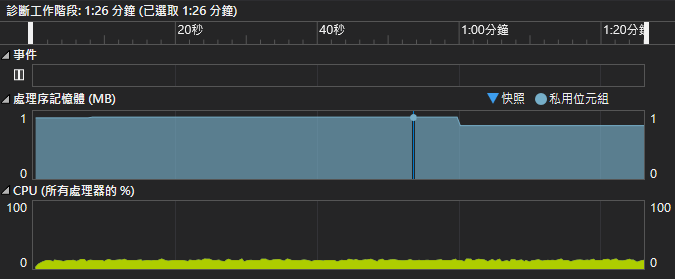
ID Number: 11363111 Name: 陳瑞鑫

|  |  |
| --- | --- |
| Program File | **Image\_Huffman.cpp** |
| Encode Input File | **lenna.bmp:** 　　512 \* 512, gray, bmp file. |
| Encode Output File | **bmpHeader.txt:** 　　Binary bmp file header + palette.  **huffTable.txt:** 　　0 – 255 Huffman code. Store by string.  **lennaCompression.txt:**  　　Binary encode bmp array. |
| Decode Input File | **bmpHeader.txt:** 　　Binary bmp file header + palette.  **huffTable.txt:** 　　0 – 255 Huffman code. Store by string.  **lennaCompression.txt:**  　　Binary encode bmp array. |
| Decode Output File | **lenna\_r.bmp**: 　　Decode from huffTable.txt, lennaCompression.txt.  　　Header equal lenna.bmp. |
| Time | **Debug mode:** About 90 Seconds (contain decode and encode).  **Release mode**: About 5 Seconds (contain decode and encode). |
| Memory (Debug mode) | About 1 MB. |
| CPU | Intel Core I7-9700K |
| RAM | 16 GB |

Debug mode:



Q1: Use C/C++ implement a Huffman coder(decoder)

Please reference Image\_Huffman.cpp.

Q2: Use Encoder to encode lenna.bmp, and calculate bits per pixel

Please reference output



Q3: Decode to lenna\_r.bmp, and calculate lenna.bmp and lenna\_r.bmp MSE (Mean-Square error)

Please reference output



Q4: YouTube URL:

<https://www.youtube.com/watch?v=qH4l8KlRuRM>

Timeline(**YouTube影片說明欄** / Back up content from video descriptions):

0:00 展示程式執行過程 / Demo

0:10 計算題 / bits per pixel and MSE (Mean-Square Error)

0:25 比較 lenna.bmp 跟 lenna\_r.bmp / Show lenna.bmp and lenna\_r.bmp HEX

0:40 Show bmpHeader.txt, huffTable.txt, lennaCompression.txt

1:14 Show lenna.bmp and lenna\_r.bmp

* Execute:
  + cd “Image\_Huffman/**Release**”
  + Make sure contain “lenna.bmp” and “Image\_Huffman.exe”  
    
  + Execute Image\_Huffman.exe

