**알고리즘 HW\_Huffman\_Decompression**

**2016706033 이동준**

**1.개발환경**

본 과제는 visual studio 2019에서 개발되었습니다.

**2.huffman 디코더 체계**

인코더에서 압축했던 바이너리 파일을 1Byte씩 읽어온다.

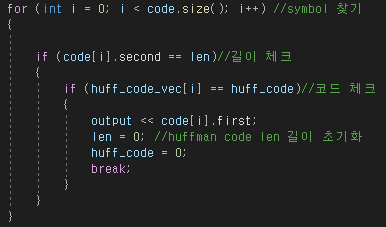
**2-1 비트스트림 구조**

비트스트림의구조는 아래와같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Huffman code symbol의갯수 | 마지막 비트스트림의 유효 비트수 | Ascii code | Huffman code 길이 | 압축된 바이너리 파일 |
| 1Byte | 1Byte | Symbol당 1Byte | Symbol당 1Byte |  |

**2-2 Huffman code로부터 압축해제**

압축된 바이너리 파일을 1Byte씩 읽어와 huffman tree에 있는 huffman code에 맞게 복호화를 시작한다. 그 C code는 아래와 같다.

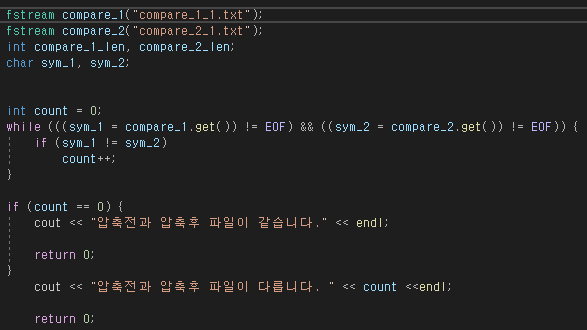


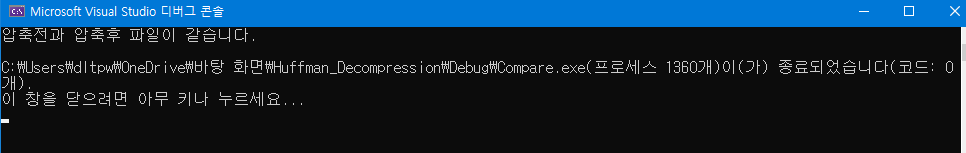
Huffman code 길이, 코드를 체크하고 그에 맞는 symbol을 저장한다.

**2-3 마지막 비트스트림의 유효비트수**

파일의 끝에서 마지막 비트스트림을 읽어왔을때, 거기서 실제로 쓰이는 비트의 수를 알아야 정확히 압축을 해제할수있다. 그 이유는 비트스트림은 무조건 8비트씩 읽어오기 때문에 필요없는 비트 ‘0’을 같이 읽어온다. 따라서 이러한 비트를 제외하고 제대로 정보만을 읽어오기 위해서 유효비트를 맨처음 해더에 저장했고 이 값을 이용해 마지막 비트스트림을 읽어온다.

**3.실험결과**





길이가 서로 다른 5개의 파일쌍으로 실험해본 결과 압축과 압축해제가 잘되는것으로 확인되었다.

**4. 개선된점**

이전 압축기에선 유효비트수를 넣지 않아 마지막 부분에서 txt파일이 잘못 압축해제가되는 경우가 있었다. 이러한 문제를 해결하기위해 압축기의 헤더에 유효비트를 저장하도록 수정했다.