Huffman Encoding/Decoding

201801730 김다희

#4

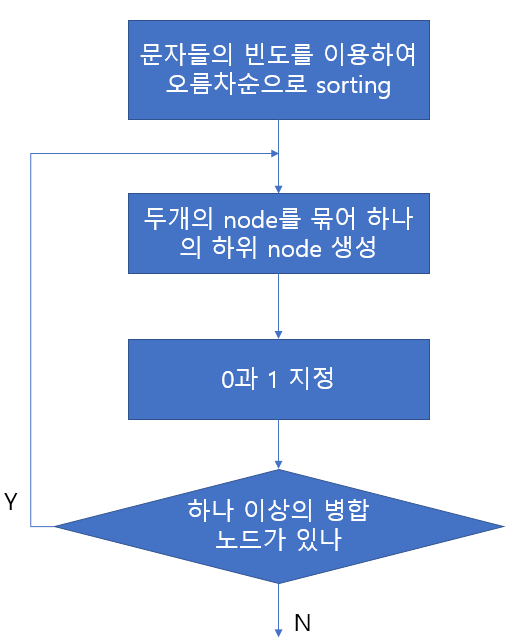
<알고리즘>

-encoding

파일 입출력을 이용하여 파일을 읽어온 후, 빈도수를 계산한다.

빈도수가 1이상인 경우, 노드를 생성해준다.

두개의 노드를 가져와 새로운 노드를 생성하여 Huffman tree를 구현한다.



Tree에 0과1을 표시하여 2진수로 표현이 가능하게 만들어준다.

Binary file을 저장한 후, file의 전체 비트의 개수의 파일을 읽어와 저장하는 비트의 올바른 위치에 넣기 위해 shift연산을 한다.

그 다음, ORmasking으로 data을 유지하며 data추가

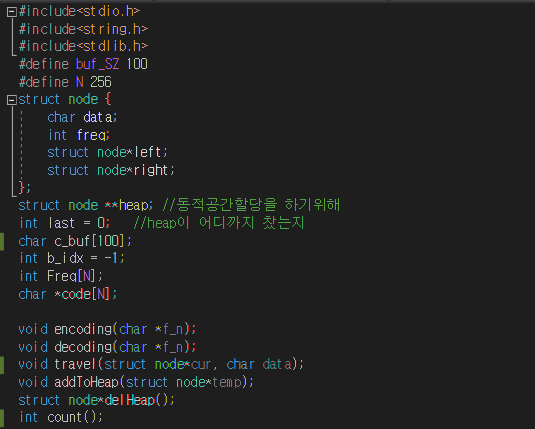
-decoding

encoding한 binary file을 읽어온 후, Huffman tree를 복구시킨다.

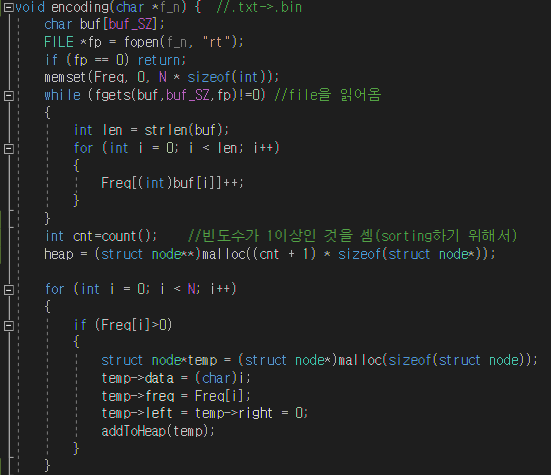
복구시킨 tree를 이용하여 Huffman code를 읽어온다.

그 값을 ANDmasking을 이용하여 2진수의 data를 가져온 뒤, 문자로 다시 출력한다.

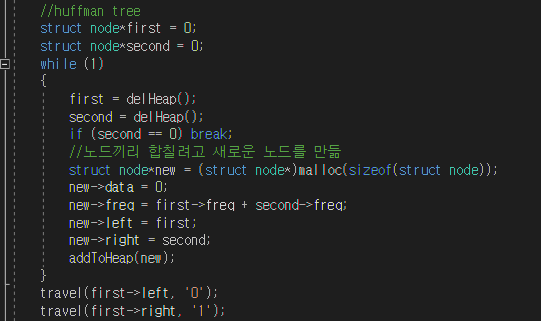
<코드>



헤더파일, 구조체, 전역변수, 함수들 선언한다.

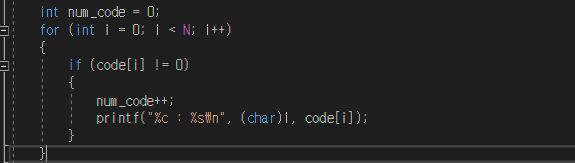


파일을 읽어온 후, 파일 안에 있는 문자들의 빈도수를 계산한다.



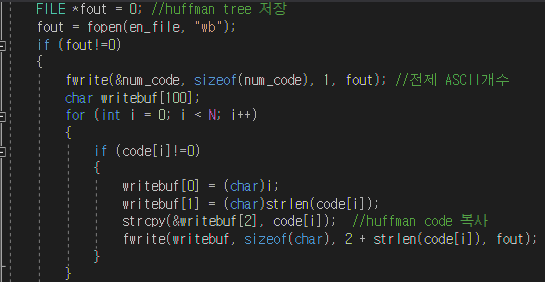
Huffman tree를 구현하고 tree를 타고 갈 때 왼쪽에는 left 오른쪽에는 right를 부여 후, 합친 노드의 새로운 parent노드를 만든다.

Travel를 이용해 0과 1을 부여한다.

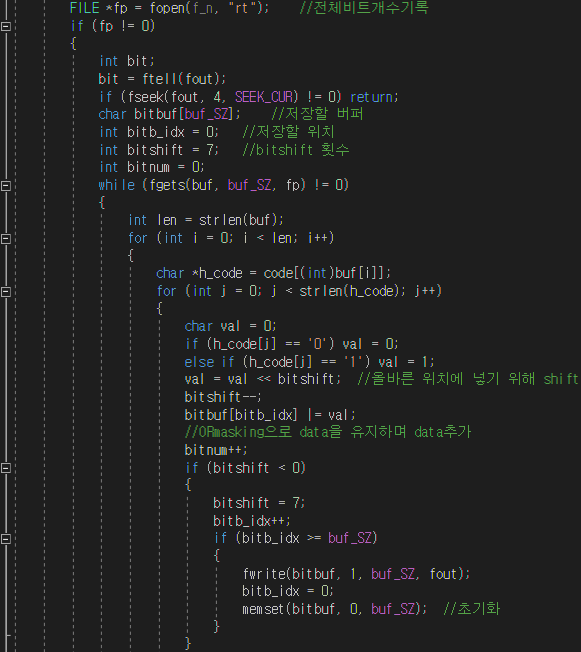


ASCII 문자의 개수를 저장한다.

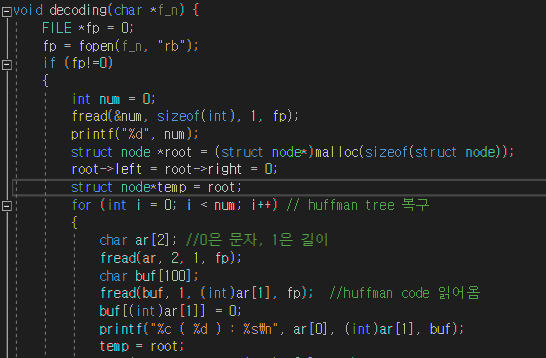
이진수로 바꾼 data를 binary file로 저장한다.



파일의 ASCII code의 개수만큼 Huffman tree를 binary file로 저장한 후, Huffman code를 미리 만들어 놓은 배열에 복사한다.

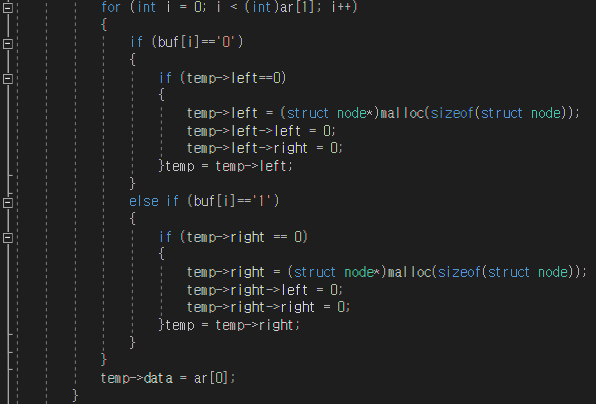


Binary file을 저장한 후, file의 전체 비트의 개수의 파일을 읽어와 저장하는 비트의 올바른 위치에 넣기 위해 shift연산을 한다.

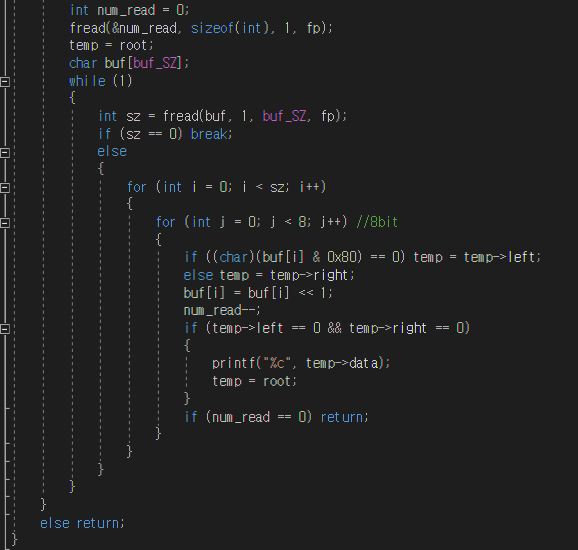


Binary file을 읽어온 후, 반복문을 이용하여 Huffman tree를 복구한다.

반복문안에서 fread를 이용하여 Huffman code를 읽어온다.

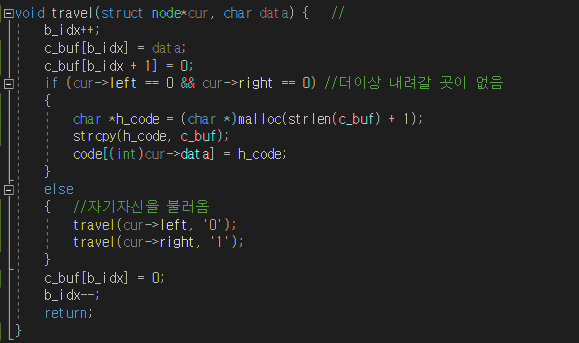


배열의 data가 0 or 1인가에 따라 tree의 왼쪽 or 오른쪽으로 따라 내려간다.

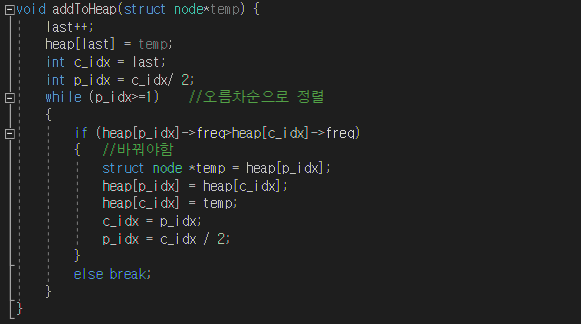


1바이트는 8비트이기 때문에 반복문을 8번 돌려, 11110000과 ANDmasking을 하여 tree을 내려갈 왼쪽과 오른쪽을 정한다.

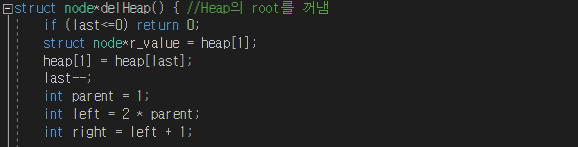
만약 temp노드의 왼쪽과 오른쪽이 0이라면 ASCII 문자를 출력하고, 다시 (root노드) 처음으로 돌아간다.



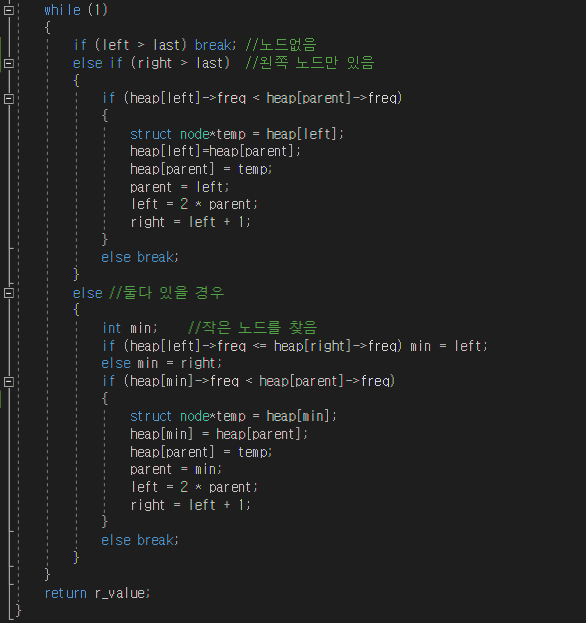
Tree를 타고 내려가며 더 이상 내려갈 곳이 없을 때까지 자기자신을 불러온다.



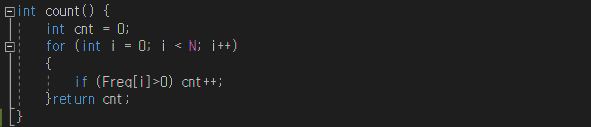
오름차순으로 sorting하기 위해서 heap에 들어있는 빈도수를 비교하여, 서로의 위치를 바꿔나가 heap을 추가한다.



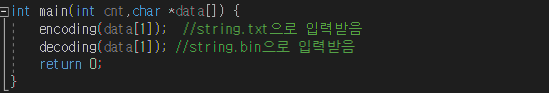
빈도수가 작은 2개의 노드를 합쳐나가 Huffman tree를 만들기 위해, heap의 root를 꺼낸다.



노드가 없는 경우, 왼쪽 노드만 있는 경우, 둘다 노드가 있는 경우로 나누어 빈도수를 비교하여 sorting과정을 실행한다.



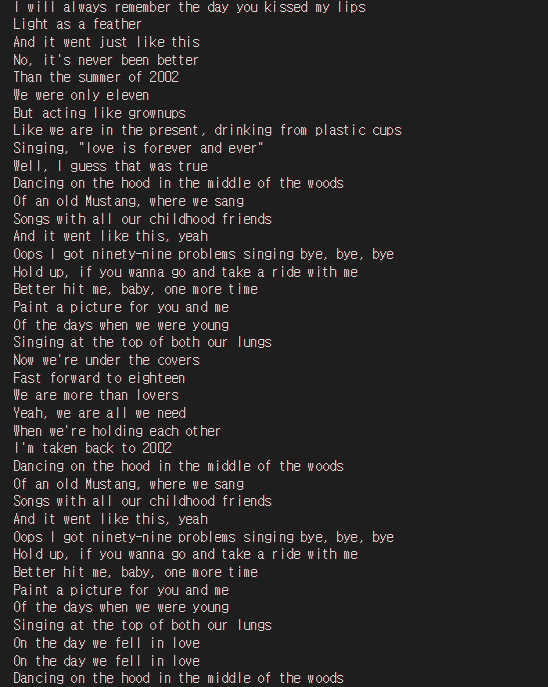
ASCII code의 빈도수를 센다.

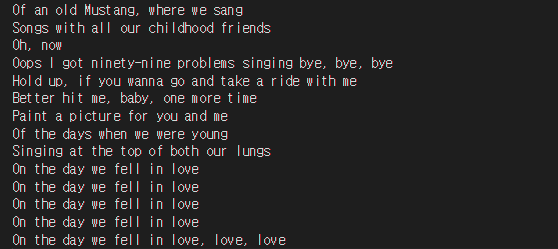


Encoding를 할지, decoding을 할지 결정한다.

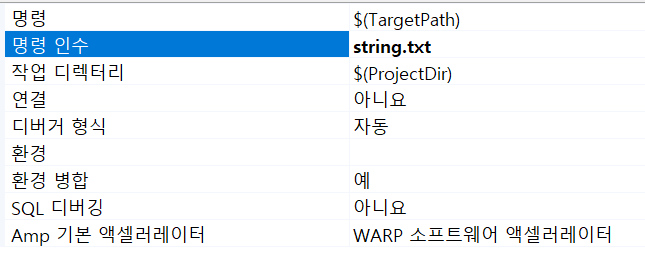
<결과>

-input txt file

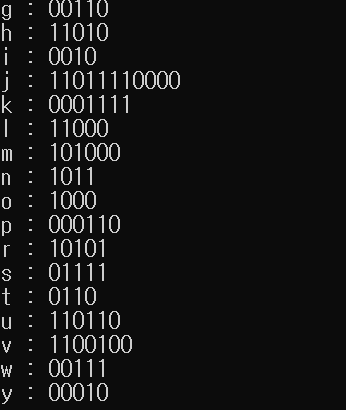
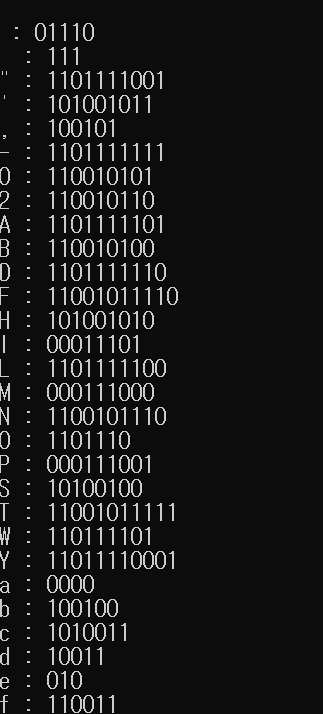




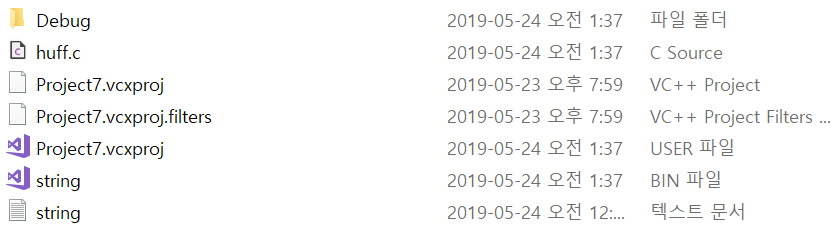
-encoding



Project의 input을 string.txt로 설정한다.



빈도수를 기반으로 table을 만든 후, travel함수를 이용하여 tree의 왼쪽 or 오른쪽으로 내려가는 지를 0과1로 표현한 값을 출력한다.

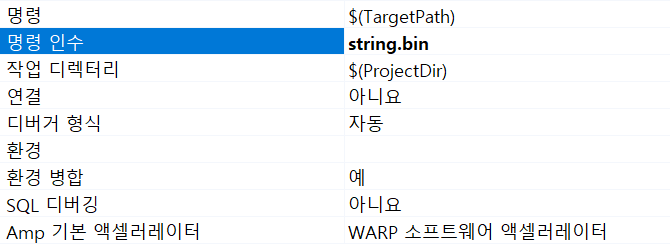


String.txt을 encoding한 data를 file로 저장하면 이와 같이 string.bin이라는 별도의 binary file이 생성된다.

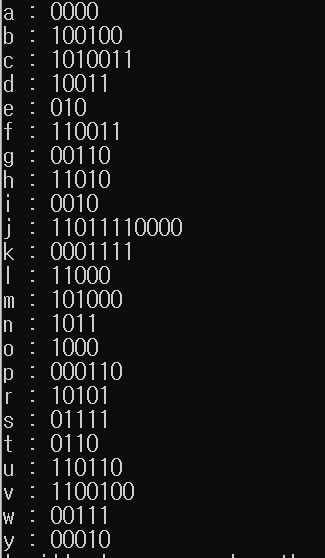


<string.bin file>

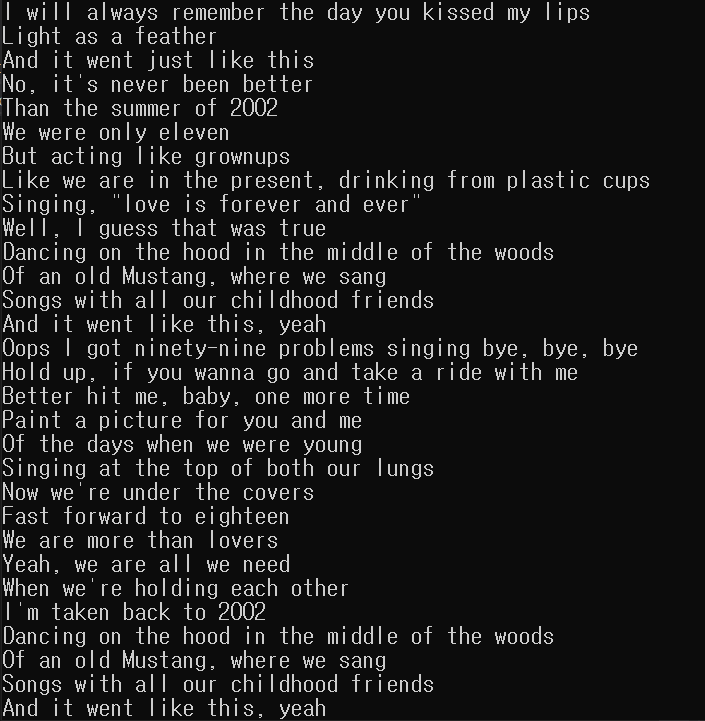
-decoding

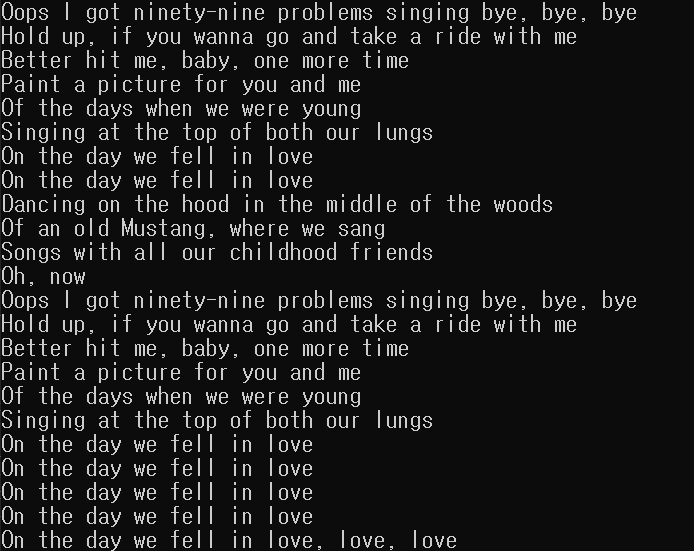


Project의 input을 string.bin으로 설정한다.



Huffman tree를 복구한 값을 출력한다.





복구한 tree를 이용하여 huffman code를 ASCII 문자로 바꾼 data를 출력한다.