import heapq

from collections import defaultdict, Counter

# Define a Node class for the Huffman tree

class Node:

def \_\_init\_\_(self, char, freq):

self.char = char

self.freq = freq

self.left = None

self.right = None

def \_\_lt\_\_(self, other):

return self.freq < other.freq

# Function to build the Huffman tree

def build\_huffman\_tree(text):

char\_freq = Counter(text)

heap = [Node(char, freq) for char, freq in char\_freq.items()]

heapq.heapify(heap)

while len(heap) > 1:

left = heapq.heappop(heap)

right = heapq.heappop(heap)

merged = Node(None, left.freq + right.freq)

merged.left = left

merged.right = right

heapq.heappush(heap, merged)

return heap[0]

# Function to generate Huffman codes

def generate\_huffman\_codes(end, current\_code, huffman\_codes):

if end is None:

return

if end.char is not None:

huffman\_codes[end.char] = current\_code

return

generate\_huffman\_codes(end.left, current\_code + '0', huffman\_codes)

generate\_huffman\_codes(end.right, current\_code + '1', huffman\_codes)

# Function to encode text using Huffman coding

def huffman\_encode(text):

end = build\_huffman\_tree(text)

huffman\_codes = {}

generate\_huffman\_codes(end, '', huffman\_codes)

encoded\_text = ''.join(huffman\_codes[char] for char in text)

return encoded\_text, huffman\_codes

# Function to decode text using Huffman coding

def huffman\_decode(encoded\_text, huffman\_codes):

reverse\_huffman\_codes = {code: char for char, code in huffman\_codes.items()}

decoded\_text = ''

current\_code = ''

for bit in encoded\_text:

current\_code += bit

if current\_code in reverse\_huffman\_codes:

char = reverse\_huffman\_codes[current\_code]

decoded\_text += char

current\_code = ''

return decoded\_text

# Main program

if \_\_namewith\_\_ == "\_\_main\_\_":

input\_text = "Your input text here" # Replace with your input text

print("Input Text:")

print(input\_text)

encoded\_text, huffman\_codes = huffman\_encode(input\_text)

print("\nEncoded Text:")

print(encoded\_text)

decoded\_text = huffman\_decode(encoded\_text, huffman\_codes)

print("\nDecoded Text:")

print(decoded\_text)