[과제연구] 최종 TTS 코드

```
import tensorflow as tf
import os
import cv2
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from numpy import array
from gtts import gTTS
import re

###학습 데이터 가공
#학습데이터 가져올 위치
TRAIN_DIR = 'D:/deeplearning/programming/fingerlanguage/traindata'
train_folder_list = array(os.listdir(TRAIN_DIR))

#데이터값들은 input 에, 정답은 label에
train_input = []
train_label = []

output_nodes = 31

label_encoder = LabelEncoder()
#폴더들을 인식, 그것들의 이름을 순차적으로 숫자형 데이터로 전환하기
integer_encoded = label_encoder.fit_transform(train_folder_list)

onehot_encoder = OneHotEncoder(sparse=False)
#onehotencoder 쓰기 위해서 integer_encoded 를 2차원배열으로 전환한다.
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
#해당 값을 가지는 index만 값을 1로 하고 나머지는 다 0을 가지는 배열으로 전환한다.
onehot_encoded = onehot_encoder.fit_transform(integer_encoded)

#img를 변환시키는 과정
for index in range(len(train_folder_list)):
    path = os.path.join(TRAIN_DIR, train_folder_list[index])
    path = path + '/'
    #경로에 / 를 추가한다.
    img_list = os.listdir(path)
    for img in img_list:
        # 해당 경로 내 폴더들에 대해서 이미지를 하나씩 읽어준다.
```

```
        img_path = os.path.join(path, img)
        #print(img_path)
        #이미지를 읽어온다.
        img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
        #print(img)
        #이미지를 numpy형 배열으로 변환해서 train_input에 넣는다.
        train_input.append([np.array(img)])
        #라벨 (정답) 도 마찬가지로 train_label에 넣는다.
        train_label.append([np.array(onehot_encoded[index])])

#input 데이터를 784개 (28x28) 가 있는 배열으로 변환
#-1은 데이터 개수 정확히 모를 때 사용하는 거
#label 데이터도 마찬가지
train_input = np.reshape(train_input, (-1, 784))
train_label = np.reshape(train_label, (-1, output_nodes))
#최종 변환
train_input = np.array(train_input).astype(np.float32)
train_label = np.array(train_label).astype(np.float32)
#print(train_label)
#이거를 npy 파일로 저장 (실제로 해당폴더에 저장됨)
np.save("train_data.npy", train_input)
np.save("train_label.npy", train_label)

###테스트 데이터 가공
#학습 데이터와 같은 원리
TEST_DIR = 'D:/deeplearning/programming/fingerlanguage/testdata'
test_folder_list = array(os.listdir(TEST_DIR))

test_input = []
test_label = []

label_encoder2 = LabelEncoder()
integer_encoded2 = label_encoder.fit_transform(test_folder_list)

onehot_encoder2 = OneHotEncoder(sparse=False)
integer_encoded2 = integer_encoded2.reshape(len(integer_encoded2), 1)
onehot_encoded2 = onehot_encoder2.fit_transform(integer_encoded2)

path = ''
```

[과제연구] 최종 TTS 코드

```
for index in range(len(test_folder_list)):
    path = os.path.join(TEST_DIR, test_folder_list[index])
    path = path + '/'
    img_list = os.listdir(path)
    for img in img_list:
        img_path = os.path.join(path, img)
        img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
        test_input.append([np.array(img)])
        test_label.append([np.array(onehot_encoded2[index])])

test_input = np.reshape(test_input, (-1, 784))
test_label = np.reshape(test_label, (-1, output_nodes))
test_input = np.array(test_input).astype(np.float32)
test_label = np.array(test_label).astype(np.float32)
np.save("test_input.npy", test_input)
np.save("test_label.npy", test_label)

# 실제학습 시작
X = tf.placeholder(tf.float32, [None, 784])
X_img = tf.reshape(X, [-1, 28, 28, 1])
Y = tf.placeholder(tf.float32, [None, output_nodes])  # 출력값
# 추후 드롭아웃 시 사용할 변수 keep_prob
keep_prob = tf.placeholder(tf.float32)

# 학습률 설정
learning_rate = 0.001

# 컨볼루션 계층 생성하기(3x3) (내장함수사용)
W1 = tf.Variable(tf.random_normal([3, 3, 1, 32], stddev=0.01))  # 3x3 크기 커널 가진 계층, 오른쪽/아래쪽으로 1칸씩 움직이며, 32개 커널
# 랜덤으로 가중치를 넣는다.
L1 = tf.nn.conv2d(X_img, W1, strides=[1, 1, 1, 1], padding='SAME')  # padding SAME : 이미지 외각 (테두리) 까지도 조금 더 명확히 평가가능
L1 = tf.nn.relu(L1)  # 활성화함수로 relu 적용

# 풀링 계층 (내장함수사용) (맥스풀링사용)
L1 = tf.nn.max_pool(L1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')

# 두 번째 계층 역시 같은 방법으로
```

[과제연구] 최종 TTS 코드

```
W2 = tf.Variable(tf.random_normal([3, 3, 32, 64], stddev=0.01))  # L1 풀링 맵을 기본으로 (32개) -> #이번에는 3x3 의 커널 64개로 구성됨
# 랜덤으로 가중치를 넣는다.
L2 = tf.nn.conv2d(L1, W2, strides=[1, 1, 1, 1], padding='SAME')
L2 = tf.nn.relu(L2)
L2 = tf.nn.max_pool(L2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')

# 10개의 분류를 만들어내는 계층
W3 = tf.Variable(tf.random_normal([7 * 7 * 64, 256], stddev=0.01))
L3 = tf.reshape(L2, [-1, 7 * 7 * 64])  # 직전 폴링 계층 크기가 7x7x64, 이를 1차원으로 만든다.
L3 = tf.matmul(L3, W3)
L3 = tf.nn.relu(L3)  # 활성화함수 적용
L3 = tf.nn.dropout(L3, keep_prob)  # 과적합 방지
# 편향은 일단제거
# b = tf.Variable(tf.random_normal([4]))

W4 = tf.Variable(tf.random_normal([256, output_nodes], stddev=0.01))
model = tf.matmul(L3, W4)  # + b

cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits_v2(logits=model, labels=Y))
# 교차 엔트로피 함수
optimizer = tf.train.AdamOptimizer(learning_rate).minimize(cost)
if optimizer == 0: optimizer = optimizer + 1
# optimizer = tf.train.RMSPropOptimizer(0.001, 0.9).minimize(cost) 이거로도 가능..

init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)  # 초기화 실행시키기

batch_size = 100  # 미니배치 사용 (100개)
total_batch = int(len(train_input) / batch_size)
all_epoch = 55
before_cost = 0

print('학습 시작!')
for epoch in range(all_epoch):
    total_cost = 0
    for i in range(total_batch):
        start = ((i + 1) * batch_size) - batch_size
```

[과제연구] 최종 TTS 코드

```
        end = ((i + 1) * batch_size)  # 학습할 데이터를 배치 크기만큼 가져온 뒤,
        batch_xs = train_input[start:end]
        batch_ys = train_label[start:end]
        # batch_xs, batch_ys 정의해주기
        # 입력값 = batch_xs, 출력값 = batch_ys

        _, cost_val = sess.run([optimizer, cost], feed_dict={X: batch_xs, Y: batch_ys, keep_prob: 0.57})
        # 최적화+손실값 저장, 입력값X와 평가할 때 쓸 실제값 Y를 넣어준다. 과적합방지 비율 = 57%
        total_cost = total_cost + cost_val
        # 오차값 구하기
    print('Epoch:', '%04d' % (epoch + 1), 'Avg. cost = ', '{:3f}'.format(total_cost / total_batch))
    if before_cost == total_cost / total_batch: total_cost = total_cost + 1
    before_cost = total_cost / total_batch

print('학습 완료!')

#결과확인
is_correct = tf.equal(tf.argmax(model, 1), tf.argmax(Y, 1)) #1번 인덱스의 차원 중 최댓값의 인덱스를 뽑아낸다! equal 로 같은지 비교!
accuracy = tf.reduce_mean(tf.cast(is_correct, tf.float32)) #cast를 통해서 0과 1로 변환
print('정확도:', sess.run(accuracy, feed_dict = {X: test_input, Y: test_label, keep_prob : 1}))
result = sess.run(model, feed_dict = {X: test_input, Y: test_label, keep_prob : 1})
#print(test_input)
#각각 model 에서 얻은 결론값에 해당하는 한글 조성을 출력한다.
#현재 index 0 = ㄱ 1 = ㄹ 2 = ㅍ 3 = ㅎ
char = {0 : 'ㄱ', 1 : 'ㄴ', 2 : 'ㄷ', 3 : 'ㄹ', 4 : 'ㅁ', 5 : 'ㅂ', 6 : 'ㅅ', 7 : 'ㅇ', 8 : 'ㅈ', 9 : 'ㅊ', 10 : 'ㅋ',
        11 : 'ㅌ', 12 : 'ㅍ', 13 : 'ㅎ', 14 : 'ㅏ', 15 : 'ㅑ', 16 : 'ㅓ', 17 : 'ㅕ', 18 : 'ㅗ',
       19 : 'ㅛ', 20 : 'ㅜ', 21 : 'ㅠ', 22 : 'ㅡ', 23 : 'ㅣ', 24 : 'ㅐ', 25 : 'ㅒ', 26 : 'ㅔ', 27 : 'ㅖ', 28 : 'ㅚ',
        29 : 'ㅟ', 30 : 'ㅢ', 31 : 'x', 32 : 'x', 33 : 'x', 34 : 'x', 35 : 'x', 36 : 's'}
for i in range(len(result)):
    #print(result[i])
    maxi = -100
    ind = 0
    for j in range(0, output_nodes):
        if result[i][j] > maxi:
            maxi = result[i][j]
            ind = j
    print(char[ind])
```

[과제연구] 최종 TTS 코드

```
###TTS데이터
#학습 데이터와 같은 원리
TTS_DIR = 'D:/deeplearning/programming/fingerlanguage/ttss'

tts_input = []
img_list = os.listdir(TTS_DIR)
TTS_DIR = TTS_DIR + '/'
#경로에 / 를 추가한다.
img_list = os.listdir(TTS_DIR)
for img in img_list:
    img_path = os.path.join(TTS_DIR, img)
    img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
    tts_input.append([np.array(img)])
tts_input = np.reshape(tts_input, (-1, 784))
tts_input = np.array(tts_input).astype(np.float32)

han_result = []
#결과출력
#print(tts_input)
ttsresult = sess.run(model, feed_dict = {X: tts_input, keep_prob : 1})

#각각 model 에서 얻은 결론값에 해당하는 한글 초성을 출력한다.
for i in range(len(ttsresult)):
    maxi = -1000
    ind = 0
    for j in range(0, len(ttsresult[i])):
        if ttsresult[i][j] > maxi:
            maxi = ttsresult[i][j]
            ind = j
    ##인위적 변환
    #완료
    if ind == 6 :
        ind = ind - 5
        print(char[ind])
        han_result.append(char[ind])
        continue
    #완료
    if 25 <= ind and ind <= 30 :
        ind = ind - 22
```

```
        print(char[ind])
        han_result.append(char[ind])
        continue
    #완료
    if 1 <= ind and ind <= 5:
        ind = ind + 8
        print(char[ind])
        han_result.append(char[ind])
        continue
    #완료
    if 7 <= ind and ind <= 16:
        ind = ind + 7
        print(char[ind])
        han_result.append(char[ind])
        continue
    #완료
    if ind == 17 :
        ind = ind - 15
        print(char[ind])
        han_result.append(char[ind])
        continue
    #완료
    if 18 <= ind and ind <= 19:
        ind = ind + 6
        print(char[ind])
        han_result.append(char[ind])
        continue
    #완료
    if 20 <= ind and ind <= 24:
        ind = ind + 6
        print(char[ind])
        han_result.append( char[ind])
        continue
    print(char[ind])
    han_result.append(char[ind])
print(han_result)

#한글합성
M  =  '가각갂갃간갅굞굥교굑굒굓굔굕굖굗굘굙굚굛굜굝굞굟굠굡굢굣굤굥굦굧굨굩굪굫구국굮굯군굱굲굳굴굵굶굷굸
```

[과제연구] 최종 TTS 코드

굹굺굻굼굽굾굿궀궁궂궃궄궅궆궇궈궉궊궋권궍궎궏궐궑궒궓궔궕궖궗궘궙궚궛궜궝궞궟궠궡궢궣궤궥궦궧궨궩궪궫궬
궭궮궯궰궱궲궳궴궵궶궷궸궹궺궻궼궽궾궿귀귁귂귃귄귅귆귇귈귉귊귋귌귍귎귏귐귑귒귓귔귕귖귗귘귙귚귛규귝귞귟균
귡귢귣귤귥귦귧귨귩귪귫귬귭귮귯귰귱귲귳귴귵귶귷그극귺귻근귽귾귿글긁긂긃긄긅긆긇금급긊긋긌긍긎긏긐긑긒긓긔
긕긖긗긘긙긚긛긜긝긞긟긠긡긢긣긤긥긦긧긨긩긪긫긬긭긮긯기긱긲긳긴긵긶긷길긹긺긻긼긽긾긿김깁깂깃깄깅깆깇깈
깉깊깋까깍깎깏깐깑깒깓깔깕깖깗깘깙깚깛깜깝깞깟깠깡깢깣깤깥깦깧깨깩깪깫깬깭깮깯깰깱깲깳깴깵깶깷깸깹깺깻깼
깽깾깿꺀꺁꺂꺃꺄꺅꺆꺇꺈꺉꺊꺋꺌꺍꺎꺏꺐꺑꺒꺓꺔꺕꺖꺗꺘꺙꺚꺛꺜꺝꺞꺟꺠꺡꺢꺣꺤꺥꺦꺧꺨꺩꺪꺫꺬꺭꺮꺯꺰
꺱꺲꺳꺴꺵꺶꺷꺸꺹꺺꺻꺼꺽꺾꺿껀껁껂껃껄껅껆껇껈껉껊껋껌껍껎껏껐껑껒껓껔껕껖껗께껙껚껛껜껝껞껟껠껡껢껣껤
껥껦껧껨껩껪껫껬껭껮껯껰껱껲껳껴껵껶껷껸껹껺껻껼껽껾껿꼀꼁꼂꼃꼄꼅꼆꼇꼈꼉꼊꼋꼌꼍꼎꼏꼐꼑꼒꼓꼔꼕꼖꼗꼘
꼙꼚꼛꼜꼝꼞꼟꼠꼡꼢꼣꼤꼥꼦꼧꼨꼩꼪꼫꼬꼭꼮꼯꼰꼱꼲꼳꼴꼵꼶꼷꼸꼹꼺꼻꼼꼽꼾꼿꽀꽁꽂꽃꽄꽅꽆꽇꽈꽉꽊꽋꽌
꽍꽎꽏꽐꽑꽒꽓꽔꽕꽖꽗꽘꽙꽚꽛꽜꽝꽞꽟꽠꽡꽢꽣꽤꽥꽦꽧꽨꽩꽪꽫꽬꽭꽮꽯꽰꽱꽲꽳꽴꽵꽶꽷꽸꽹꽺꽻꽼꽽꽾꽿꾀
꾁꾂꾃꾄꾅꾆꾇꾈꾉꾊꾋꾌꾍꾎꾏꾐꾑꾒꾓꾔꾕꾖꾗꾘꾙꾚꾛꾜꾝꾞꾟꾠꾡꾢꾣꾤꾥꾦꾧꾨꾩꾪꾫꾬꾭꾮꾯꾰꾱꾲꾳꾴
꾵꾶꾷꾸꾹꾺꾻꾼꾽꾾꾿꿀꿁꿂꿃꿄꿅꿆꿇꿈꿉꿊꿋꿌꿍꿎꿏꿐꿑꿒꿓꿔꿕꿖꿗꿘꿙꿚꿛꿜꿝꿞꿟꿠꿡꿢꿣꿤꿥꿦꿧꿨
꿩꿪꿫꿬꿭꿮꿯꿰꿱꿲꿳꿴꿵꿶꿷꿸꿹꿺꿻꿼꿽꿾꿿뀀뀁뀂뀃뀄뀅뀆뀇뀈뀉뀊뀋뀌뀍뀎뀏뀐뀑뀒뀓뀔뀕뀖뀗뀘뀙뀚뀛뀜
뀝뀞뀟뀠뀡뀢뀣뀤뀥뀦뀧뀨뀩뀪뀫뀬뀭뀮뀯뀰뀱뀲뀳뀴뀵뀶뀷뀸뀹뀺뀻뀼뀽뀾뀿끀끁끂끃끄끅끆끇끈끉끊끋끌끍끎끏끐
끑끒끓끔끕끖끗끘끙끚끛끜끝끞끟끠끡끢끣끤끥끦끧끨끩끪끫끬끭끮끯끰끱끲끳끴끵끶끷끸끹끺끻끼끽끾끿낀낁낂낃낄
낅낆낇낈낉낊낋낌낍낎낏낐낑낒낓낔낕낖낗나낙낚낛난낝낞낟날낡낢낣낤낥낦낧남납낪낫났낭낮낯낰낱낲낳내낵낶낷낸
낹낺낻낼낽낾낿냀냁냂냃냄냅냆냇냈냉냊냋냌냍냎냏냐냑냒냓냔냕냖냗냘냙냚냛냜냝냞냟냠냡냢냣냤냥냦냧냨냩냪냫냬
냭냮냯냰냱냲냳냴냵냶냷냸냹냺냻냼냽냾냿넀넁넂넃넄넅넆넇너넉넊넋넌넍넎넏널넑넒넓넔넕넖넗넘넙넚넛넜넝넞넟넠
넡넢넣네넥넦넧넨넩넪넫넬넭넮넯넰넱넲넳넴넵넶넷넸넹넺넻넼넽넾넿녀녁녂녃년녅녆녇녈녉녊녋녌녍녎녏념녑녒녓녔
녕녖녗녘녙녚녛녜녝녞녟녠녡녢녣녤녥녦녧녨녩녪녫녬녭녮녯녰녱녲녳녴녵녶녷노녹녺녻논녽녾녿놀놁놂놃놄놅놆놇놈
놉놊놋놌농놎놏놐놑높놓놔놕놖놗놘놙놚놛놜놝놞놟놠놡놢놣놤놥놦놧놨놩놪놫놬놭놮놯놰놱놲놳놴놵놶놷놸놹놺놻놼
놽놾놿뇀뇁뇂뇃뇄뇅뇆뇇뇈뇉뇊뇋뇌뇍뇎뇏뇐뇑뇒뇓뇔뇕뇖뇗뇘뇙뇚뇛뇜뇝뇞뇟뇠뇡뇢뇣뇤뇥뇦뇧뇨뇩뇪뇫뇬뇭뇮뇯뇰
뇱뇲뇳뇴뇵뇶뇷뇸뇹뇺뇻뇼뇽뇾뇿눀눁눂눃누눅눆눇눈눉눊눋눌눍눎눏눐눑눒눓눔눕눖눗눘눙눚눛눜눝눞눟눠눡눢눣눤
눥눦눧눨눩눪눫눬눭눮눯눰눱눲눳눴눵눶눷눸눹눺눻눼눽눾눿뉀뉁뉂뉃뉄뉅뉆뉇뉈뉉뉊뉋뉌뉍뉎뉏뉐뉑뉒뉓뉔뉕뉖뉗뉘
뉙뉚뉛뉜뉝뉞뉟뉠뉡뉢뉣뉤뉥뉦뉧뉨뉩뉪뉫뉬뉭뉮뉯뉰뉱뉲뉳뉴뉵뉶뉷뉸뉹뉺뉻뉼뉽뉾뉿늀늁늂늃늄늅늆늇늈늉늊늋늌
늍늎늏느늑늒늓는늕늖늗늘늙늚늛늜늝늞늟늠늡늢늣늤능늦늧늨늩늪늫늬늭늮늯늰늱늲늳늴늵늶늷늸늹늺늻늼늽늾늿닀
닁닂닃닄닅닆닇니닉닊닋닌닍닎닏닐닑닒닓닔닕닖닗님닙닚닛닜닝닞닟닠닡닢닣다닥닦닧단닩닪닫달닭닮닯닰닱닲닳담
답닶닷닸당닺닻닼닽닾닿대댁댂댃댄댅댆댇댈댉댊댋댌댍댎댏댐댑댒댓댔댕댖댗댘댙댚댛댜댝댞댟댠댡댢댣댤댥댦댧댨
댩댪댫댬댭댮댯댰댱댲댳댴댵댶댷댸댹댺댻댼댽댾댿덀덁덂덃덄덅덆덇덈덉덊덋덌덍덎덏덐덑덒덓더덕덖덗던덙덚덛덜
덝덞덟덠덡덢덣덤덥덦덧덨덩덪덫덬덭덮덯데덱덲덳덴덵덶덷델덹덺덻덼덽덾덿뎀뎁뎂뎃뎄뎅뎆뎇뎈뎉뎊뎋뎌뎍뎎뎏뎐
뎑뎒뎓뎔뎕뎖뎗뎘뎙뎚뎛뎜뎝뎞뎟뎠뎡뎢뎣뎤뎥뎦뎧뎨뎩뎪뎫뎬뎭뎮뎯뎰뎱뎲뎳뎴뎵뎶뎷뎸뎹뎺뎻뎼뎽뎾뎿돀돁돂돃도
독돆돇돈돉돊돋돌돍돎돏돐돑돒돓돔돕돖돗돘동돚돛돜돝돞돟돠돡돢돣돤돥돦돧돨돩돪돫돬돭돮돯돰돱돲돳돴돵돶돷돸
돹돺돻돼돽돾돿됀됁됂됃됄됅됆됇됈됉됊됋됌됍됎됏됐됑됒됓됔됕됖됗되됙됚됛된됝됞됟될됡됢됣됤됥됦됧됨됩됪됫됬
됭됮됯됰됱됲됳됴됵됶됷됸됹됺됻됼됽됾됿둀둁둂둃둄둅둆둇둈둉둊둋둌둍둎둏두둑둒둓둔둕둖둗둘둙둚둛둜둝둞둟둠
둡둢둣둤둥둦둧둨둩둪둫둬둭둮둯둰둱둲둳둴둵둶둷둸둹둺둻둼둽둾둿뒀뒁뒂뒃뒄뒅뒆뒇뒈뒉뒊뒋뒌뒍뒎뒏뒐뒑뒒뒓뒔
뒕뒖뒗뒘뒙뒚뒛뒜뒝뒞뒟뒠뒡뒢뒣뒤뒥뒦뒧뒨뒩뒪뒫뒬뒭뒮뒯뒰뒱뒲뒳뒴뒵뒶뒷뒸뒹뒺뒻뒼뒽뒾뒿듀듁듂듃듄듅듆듇듈
듉듊듋듌듍듎듏듐듑듒듓듔듕듖듗듘듙듚듛드득듞듟든듡듢듣들듥듦듧듨듩듪듫듬듭듮듯듰등듲듳듴듵듶듷듸듹듺듻듼
듽듾듿딀딁딂딃딄딅딆딇딈딉딊딋딌딍딎딏딐딑딒딓디딕딖딗딘딙딚딛딜딝딞딟딠딡딢딣딤딥딦딧딨딩딪딫딬딭딮딯따
딱딲딳딴딵딶딷딸딹딺딻딼딽딾딿땀땁땂땃땄땅땆땇땈땉땊땋때땍땎땏땐땑땒땓땔땕땖땗땘땙땚땛땜땝땞땟땠땡땢땣땤

[과제연구] 최종 TTS 코드

땥땦땧땨땩땪땫땬땭땮땯땰땱땲땳땴땵땶땷땸땹땺땻땼땽땾땿떀떁떂떃떄떅떆떇떈떉떊떋떌떍떎떏떐떑떒떓떔떕떖떗떘
떙떚떛떜떝떞떟떠떡떢떣떤떥떦떧떨떩떪떫떬떭떮떯떰떱떲떳떴떵떶떷떸떹떺떻떼떽떾떿뗀뗁뗂뗃뗄뗅뗆뗇뗈뗉뗊뗋뗌
뗍뗎뗏뗐뗑뗒뗓뗔뗕뗖뗗뗘뗙뗚뗛뗜뗝뗞뗟뗠뗡뗢뗣뗤뗥뗦뗧뗨뗩뗪뗫뗬뗭뗮뗯뗰뗱뗲뗳뗴뗵뗶뗷뗸뗹뗺뗻뗼뗽뗾뗿똀
똁똂똃똄똅똆똇똈똉똊똋똌똍똎똏또똑똒똓똔똕똖똗똘똙똚똛똜똝똞똟똠똡똢똣똤똥똦똧똨똩똪똫똬똭똮똯똰똱똲똳똴
똵똶똷똸똹똺똻똼똽똾똿뙀뙁뙂뙃뙄뙅뙆뙇뙈뙉뙊뙋뙌뙍뙎뙏뙐뙑뙒뙓뙔뙕뙖뙗뙘뙙뙚뙛뙜뙝뙞뙟뙠뙡뙢뙣뙤뙥뙦뙧뙨
뙩뙪뙫뙬뙭뙮뙯뙰뙱뙲뙳뙴뙵뙶뙷뙸뙹뙺뙻뙼뙽뙾뙿뚀뚁뚂뚃뚄뚅뚆뚇뚈뚉뚊뚋뚌뚍뚎뚏뚐뚑뚒뚓뚔뚕뚖뚗뚘뚙뚚뚛뚜
뚝뚞뚟뚠뚡뚢뚣뚤뚥뚦뚧뚨뚩뚪뚫뚬뚭뚮뚯뚰뚱뚲뚳뚴뚵뚶뚷뚸뚹뚺뚻뚼뚽뚾뚿뛀뛁뛂뛃뛄뛅뛆뛇뛈뛉뛊뛋뛌뛍뛎뛏뛐
뛑뛒뛓뛔뛕뛖뛗뛘뛙뛚뛛뛜뛝뛞뛟뛠뛡뛢뛣뛤뛥뛦뛧뛨뛩뛪뛫뛬뛭뛮뛯뛰뛱뛲뛳뛴뛵뛶뛷뛸뛹뛺뛻뛼뛽뛾뛿뜀뜁뜂뜃뜄
뜅뜆뜇뜈뜉뜊뜋뜌뜍뜎뜏뜐뜑뜒뜓뜔뜕뜖뜗뜘뜙뜚뜛뜜뜝뜞뜟뜠뜡뜢뜣뜤뜥뜦뜧뜨뜩뜪뜫뜬뜭뜮뜯뜰뜱뜲뜳뜴뜵뜶뜷뜸
뜹뜺뜻뜼뜽뜾뜿띀띁띂띃띄띅띆띇띈띉띊띋띌띍띎띏띐띑띒띓띔띕띖띗띘띙띚띛띜띝띞띟띠띡띢띣띤띥띦띧띨띩띪띫띬
띭띮띯띰띱띲띳띴띵띶띷띸띹띺띻라락랚랛란랝랞랟랄랅랆랇랈랉랊랋람랍랎랏랐랑랒랓랔랕랖랗래랙랚랛랜랝랞랟랠
랡랢랣랤랥랦랧램랩랪랫랬랭랮랯랰랱랲랳랴략랶랷랸랹랺랻랼랽랾랿럀럁럂럃럄럅럆럇럈량럊럋럌럍럎럏럐럑럒럓럔
럕럖럗럘럙럚럛럜럝럞럟럠럡럢럣럤럥럦럧럨럩럪럫러럭럮럯런럱럲럳럴럵럶럷럸럹럺럻럼럽럾럿렀렁렂렃렄렅렆렇레
렉렊렋렌렍렎렏렐렑렒렓렔렕렖렗렘렙렚렛렜렝렞렟렠렡렢렣려력렦렧련렩렪렫렬렭렮렯렰렱렲렳렴렵렶렷렸령렺렻렼
렽렾렿례롁롂롃롄롅롆롇롈롉롊롋롌롍롎롏롐롑롒롓롔롕롖롗롘롙롚롛로록롞롟론롡롢롣롤롥롦롧롨롩롪롫롬롭롮롯롰
롱롲롳롴롵롶롷롸롹롺롻롼롽롾롿뢀뢁뢂뢃뢄뢅뢆뢇뢈뢉뢊뢋뢌뢍뢎뢏뢐뢑뢒뢓뢔뢕뢖뢗뢘뢙뢚뢛뢜뢝뢞뢟뢠뢡뢢뢣뢤
뢥뢦뢧뢨뢩뢪뢫뢬뢭뢮뢯뢰뢱뢲뢳뢴뢵뢶뢷뢸뢹뢺뢻뢼뢽뢾뢿룀룁룂룃룄룅룆룇룈룉룊룋료룍룎룏룐룑룒룓룔룕룖룗룘
룙룚룛룜룝룞룟룠룡룢룣룤룥룦룧루룩룪룫룬룭룮룯룰룱룲룳룴룵룶룷룸룹룺룻룼룽룾룿뤀뤁뤂뤃뤄뤅뤆뤇뤈뤉뤊뤋뤌
뤍뤎뤏뤐뤑뤒뤓뤔뤕뤖뤗뤘뤙뤚뤛뤜뤝뤞뤟뤠뤡뤢뤣뤤뤥뤦뤧뤨뤩뤪뤫뤬뤭뤮뤯뤰뤱뤲뤳뤴뤵뤶뤷뤸뤹뤺뤻뤼뤽뤾뤿륀
륁륂륃륄륅륆륇륈륉륊륋륌륍륎륏륐륑륒륓륔륕륖륗류륙륚륛륜륝륞륟률륡륢륣륤륥륦륧륨륩륪륫륬륭륮륯륰륱륲륳르
륵륶륷른륹륺륻를륽륾륿릀릁릂릃름릅릆릇릈릉릊릋릌릍릎릏릐릑릒릓릔릕릖릗릘릙릚릛릜릝릞릟릠릡릢릣릤릥릦릧릨
릩릪릫리릭릮릯린릱릲릳릴릵릶릷릸릹릺릻림립릾릿맀링맂맃맄맅맆맇마막맊맋만맍많맏말맑맒맓맔맕맖맗맘맙맚맛맜
망맞맟맠맡맢맣매맥맦맧맨맩맪맫맬맭맮맯맰맱맲맳맴맵맶맷맸맹맺맻맼맽맾맿먀먁먂먃먄먅먆먇먈먉먊먋먌먍먎먏먐
먑먒먓먔먕먖먗먘먙먚먛먜먝먞먟먠먡먢먣먤먥먦먧먨먩먪먫먬먭먮먯먰먱먲먳먴먵먶먷머먹먺먻먼먽먾먿멀멁멂멃멄
멅멆멇멈멉멊멋멌멍멎멏멐멑멒멓메멕멖멗멘멙멚멛멜멝멞멟멠멡멢멣멤멥멦멧멨멩멪멫멬멭멮멯며멱멲멳면멵멶멷멸
멹멺멻멼멽멾멿몀몁몂몃몄명몆몇몈몉몊몋몌몍몎몏몐몑몒몓몔몕몖몗몘몙몚몛몜몝몞몟몠몡몢몣몤몥몦몧모목몪몫몬
몭몮몯몰몱몲몳몴몵몶몷몸몹몺못몼몽몾몿뫀뫁뫂뫃뫄뫅뫆뫇뫈뫉뫊뫋뫌뫍뫎뫏뫐뫑뫒뫓뫔뫕뫖뫗뫘뫙뫚뫛뫜뫝뫞뫟뫠
뫡뫢뫣뫤뫥뫦뫧뫨뫩뫪뫫뫬뫭뫮뫯뫰뫱뫲뫳뫴뫵뫶뫷뫸뫹뫺뫻뫼뫽뫾뫿묀묁묂묃묄묅묆묇묈묉묊묋묌묍묎묏묐묑묒묓묔
묕묖묗묘묙묚묛묜묝묞묟묠묡묢묣묤묥묦묧묨묩묪묫묬묭묮묯묰묱묲묳무묵묶묷문묹묺묻물묽묾묿뭀뭁뭂뭃뭄뭅뭆뭇뭈
뭉뭊뭋뭌뭍뭎뭏뭐뭑뭒뭓뭔뭕뭖뭗뭘뭙뭚뭛뭜뭝뭞뭟뭠뭡뭢뭣뭤뭥뭦뭧뭨뭩뭪뭫뭬뭭뭮뭯뭰뭱뭲뭳뭴뭵뭶뭷뭸뭹뭺뭻뭼
뭽뭾뭿뮀뮁뮂뮃뮄뮅뮆뮇뮈뮉뮊뮋뮌뮍뮎뮏뮐뮑뮒뮓뮔뮕뮖뮗뮘뮙뮚뮛뮜뮝뮞뮟뮠뮡뮢뮣뮤뮥뮦뮧뮨뮩뮪뮫뮬뮭뮮뮯뮰
뮱뮲뮳뮴뮵뮶뮷뮸뮹뮺뮻뮼뮽뮾뮿므믁믂믃믄믅믆믇믈믉믊믋믌믍믎믏믐믑믒믓믔믕믖믗믘믙믚믛믜믝믞믟믠믡믢믣믤
믥믦믧믨믩믪믫믬믭믮믯믰믱믲믳믴믵믶믷미믹믺믻민믽믾믿밀밁밂밃밄밅밆밇밈밉밊밋밌밍밎및밐밑밒밓바박밖밗반
밙밚받발밝밞밟밠밡밢밣밤밥밦밧밨방밪밫밬밭밮밯배백밲밳밴밵밶밷밸밹밺밻밼밽밾밿뱀뱁뱂뱃뱄뱅뱆뱇뱈뱉뱊뱋뱌
뱍뱎뱏뱐뱑뱒뱓뱔뱕뱖뱗뱘뱙뱚뱛뱜뱝뱞뱟뱠뱡뱢뱣뱤뱥뱦뱧뱨뱩뱪뱫뱬뱭뱮뱯뱰뱱뱲뱳뱴뱵뱶뱷뱸뱹뱺뱻뱼뱽뱾뱿벀
벁벂벃버벅벆벇번벉벊벋벌벍벎벏벐벑벒벓범법벖벗벘벙벚벛벜벝벞벟베벡벢벣벤벥벦벧벨벩벪벫벬벭벮벯벰벱벲벳벴
벵벶벷벸벹벺벻벼벽벾벿변볁볂볃별볅볆볇볈볉볊볋볌볍볎볏볐병볒볓볔볕볖볗볘볙볚볛볜볝볞볟볠볡볢볣볤볥볦볧볨
볩볪볫볬볭볮볯볰볱볲볳보복볶볷본볹볺볻볼볽볾볿봀봁봂봃봄봅봆봇봈봉봊봋봌봍봎봏봐봑봒봓봔봕봖봗봘봙봚봛봜
봝봞봟봠봡봢봣봤봥봦봧봨봩봪봫봬봭봮봯봰봱봲봳봴봵봶봷봸봹봺봻봼봽봾봿뵀뵁뵂뵃뵄뵅뵆뵇뵈뵉뵊뵋뵌뵍뵎뵏뵐

[과제연구] 최종 TTS 코드

[과제연구] 최종 TTS 코드

씽씾씿앀앁앂앃아악앆앇안앉않앋알앍앎앏앐앑앒앓암압앖앗았앙앚앛앜앝앞앟애액앢앣앤앥앦앧앨앩앪앫앬앭앮앯앰앱앲앳앴앵앶앷앸앹앺앻야약얚얛얀얁얂얃얄얅얆얇얈얉얊얋얌얍얎얏얐양얒얓얔얕얖얗얘얙얚얛얜얝얞얟얠얡얢얣얤얥얦얧얨얩얪얫얬얭얮얯얰얱얲얳어억얶얷언얹얺얻얼얽얾얿엀엁엂엃엄업없엇었엉엊엋엌엍엎엏에엑엒엓엔엕엖엗엘엙엚엛엜엝엞엟엠엡엢엣엤엥엦엧엨엩엪엫여역엮엯연엱엲엳열엵엶엷엸엹엺엻염엽엾엿였영옂옃옄옅옆옇예옉옊옋옌옍옎옏옐옑옒옓옔옕옖옗옘옙옚옛옜옝옞옟옠옡옢옣오옥옦옧온옩옪옫올옭옮옯옰옱옲옳옴옵옶옷옸옹옺옻옼옽옾옿와왁왂왃완왅왆왇왈왉왊왋왌왍왎왏왐왑왒왓왔왕왖왗왘왙왚왛왜왝왞왟왠왡왢왣왤왥왦왧왨왩왪왫왬왭왮왯왰왱왲왳왴왵왶왷외왹왺왻왼왽왾왿욀욁욂욃욄욅욆욇욈욉욊욋욌욍욎욏욐욑욒욓요욕욖욗욘욙욚욛욜욝욞욟욠욡욢욣욤욥욦욧욨용욪욫욬욭욮욯우욱욲욳운욵욶욷울욹욺욻욼욽욾욿움웁웂웃웄웅웆웇웈웉웊웋워웍웎웏원웑웒웓월웕웖웗웘웙웚웛웜웝웞웟웠웡웢웣웤웥웦웧웨웩웪웫웬웭웮웯웰웱웲웳웴웵웶웷웸웹웺웻웼웽웾웿윀윁윂윃위윅윆윇윈윉윊윋윌윍윎윏윐윑윒윓윔윕윖윗윘윙윚윛윜윝윞윟유육윢윣윤윥윦윧율윩윪윫윬윭윮윯윰윱윲윳윴융윶윷윸윹윺윻으윽윾윿은읁읂읃을읅읆읇읈읉읊읋음읍읎읏읐응읒읓읔읕읖읗의읙읚읛읜읝읞읟읠읡읢읣읤읥읦읧읨읩읪읫읬읭읮읯읰읱읲읳이익읶읷인읹읺읻일읽읾읿잀잁잂잃임입잆잇있잉잊잋잌잍잎잏자작잒잓잔잕잖잗잘잙잚잛잜잝잞잟잠잡잢잣잤장잦잧잨잩잪잫재잭잮잯잰잱잲잳잴잵잶잷잸잹잺잻잼잽잾잿쟀쟁쟂쟃쟄쟅쟆쟇쟈쟉쟊쟋쟌쟍쟎쟏쟐쟑쟒쟓쟔쟕쟖쟗쟘쟙쟚쟛쟜쟝쟞쟟쟠쟡쟢쟣쟤쟥쟦쟧쟨쟩쟪쟫쟬쟭쟮쟯쟰쟱쟲쟳쟴쟵쟶쟷쟸쟹쟺쟻쟼쟽쟾쟿저적젂젃전젅젆젇절젉젊젋젌젍젎젏점접젒젓젔정젖젗젘젙젚젛제젝젞젟젠젡젢젣젤젥젦젧젨젩젪젫젬젭젮젯젰젱젲젳젴젵젶젷져젹젺젻젼젽젾젿졀졁졂졃졄졅졆졇졈졉졊졋졌졍졎졏졐졑졒졓졔졕졖졗졘졙졚졛졜졝졞졟졠졡졢졣졤졥졦졧졨졩졪졫졬졭졮졯조족졲졳존졵졶졷졸졹졺졻졼졽졾졿좀좁좂좃좄종좆좇좈좉좊좋좌좍좎좏좐좑좒좓좔좕좖좗좘좙좚좛좜좝좞좟좠좡좢좣좤좥좦좧좨좩좪좫좬좭좮좯좰좱좲좳좴좵좶좷좸좹좺좻좼좽좾좿죀죁죂죃죄죅죆죇죈죉죊죋죌죍죎죏죐죑죒죓죔죕죖죗죘죙죚죛죜죝죞죟죠죡죢죣죤죥죦죧죨죩죪죫죬죭죮죯죰죱죲죳죴죵죶죷죸죹죺죻주죽죾죿준줁줂줃줄줅줆줇줈줉줊줋줌줍줎줏줐중줒줓줔줕줖줗줘줙줚줛줜줝줞줟줠줡줢줣줤줥줦줧줨줩줪줫줬줭줮줯줰줱줲줳줴줵줶줷줸줹줺줻줼줽줾줿쥀쥁쥂쥃쥄쥅쥆쥇쥈쥉쥊쥋쥌쥍쥎쥏쥐쥑쥒쥓쥔쥕쥖쥗쥘쥙쥚쥛쥜쥝쥞쥟쥠쥡쥢쥣쥤쥥쥦쥧쥨쥩쥪쥫쥬쥭쥮쥯쥰쥱쥲쥳쥴쥵쥶쥷쥸쥹쥺쥻쥼쥽쥾쥿즀즁즂즃즄즅즆즇즈즉즊즋즌즍즎즏즐즑즒즓즔즕즖즗즘즙즚즛즜증즞즟즠즡즢즣즤즥즦즧즨즩즪즫즬즭즮즯즰즱즲즳즴즵즶즷즸즹즺즻즼즽즾즿지직짂짃진짅짆짇질짉짊짋짌짍짎짏짐집짒짓짔징짖짗짘짙짚짛짜짝짞짟짠짡짢짣짤짥짦짧짨짩짪짫짬짭짮짯짰짱짲짳짴짵짶짷째짹짺짻짼짽짾짿쨀쨁쨂쨃쨄쨅쨆쨇쨈쨉쨊쨋쨌쨍쨎쨏쨐쨑쨒쨓쨔쨕쨖쨗쨘쨙쨚쨛쨜쨝쨞쨟쨠쨡쨢쨣쨤쨥쨦쨧쨨쨩쨪쨫쨬쨭쨮쨯쨰쨱쨲쨳쨴쨵쨶쨷쨸쨹쨺쨻쨼쨽쨾쨿쩀쩁쩂쩃쩄쩅쩆쩇쩈쩉쩊쩋쩌쩍쩎쩏쩐쩑쩒쩓쩔쩕쩖쩗쩘쩙쩚쩛쩜쩝쩞쩟쩠쩡쩢쩣쩤쩥쩦쩧쩨쩩쩪쩫쩬쩭쩮쩯쩰쩱쩲쩳쩴쩵쩶쩷쩸쩹쩺쩻쩼쩽쩾쩿쪀쪁쪂쪃쪄쪅쪆쪇쪈쪉쪊쪋쪌쪍쪎쪏쪐쪑쪒쪓쪔쪕쪖쪗쪘쪙쪚쪛쪜쪝쪞쪟쪠쪡쪢쪣쪤쪥쪦쪧쪨쪩쪪쪫쪬쪭쪮쪯쪰쪱쪲쪳쪴쪵쪶쪷쪸쪹쪺쪻쪼쪽쪾쪿쫀쫁쫂쫃쫄쫅쫆쫇쫈쫉쫊쫋쫌쫍쫎쫏쫐쫑쫒쫓쫔쫕쫖쫗쫘쫙쫚쫛쫜쫝쫞쫟쫠쫡쫢쫣쫤쫥쫦쫧쫨쫩쫪쫫쫬쫭쫮쫯쫰쫱쫲쫳쫴쫵쫶쫷쫸쫹쫺쫻쫼쫽쫾쫿쬀쬁쬂쬃쬄쬅쬆쬇쬈쬉쬊쬋쬌쬍쬎쬏쬐쬑쬒쬓쬔쬕쬖쬗쬘쬙쬚쬛쬜쬝쬞쬟쬠쬡쬢쬣쬤쬥쬦쬧쬨쬩쬪쬫쬬쬭쬮쬯쬰쬱쬲쬳쬴쬵쬶쬷쬸쬹쬺쬻쬼쬽쬾쬿쭀쭁쭂쭃쭄쭅쭆쭇쭈쭉쭊쭋쭌쭍쭎쭏쭐쭑쭒쭓쭔쭕쭖쭗쭘쭙쭚쭛쭜쭝쭞쭟쭠쭡쭢쭣쭤쭥쭦쭧쭨쭩쭪쭫쭬쭭쭮쭯쭰쭱쭲쭳쭴쭵쭶쭷쭸쭹쭺쭻쭼쭽쭾쭿쮀쮁쮂쮃쮄쮅쮆쮇쮈쮉쮊쮋쮌쮍쮎쮏쮐쮑쮒쮓쮔쮕쮖쮗쮘쮙쮚쮛쮜쮝쮞쮟쮠쮡쮢쮣쮤쮥쮦쮧쮨쮩쮪쮫쮬쮭쮮쮯쮰쮱쮲쮳쮴쮵쮶쮷쮸쮹쮺쮻쮼쮽쮾쮿쯀쯁쯂쯃쯄쯅쯆쯇쯈쯉쯊쯋쯌쯍쯎쯏쯐쯑쯒쯓쯔쯕쯖쯗쯘쯙쯚쯛쯜쯝쯞쯟쯠쯡쯢쯣쯤쯥쯦쯧쯨쯩쯪쯫쯬쯭쯮쯯쯰쯱쯲쯳쯴쯵쯶쯷쯸쯹쯺쯻쯼쯽쯾쯿찀찁찂찃찄찅찆찇찈찉찊찋찌찍찎찏찐찑찒찓찔찕찖찗찘찙찚찛찜찝찞찟찠찡찢찣찤찥찦찧차착찪찫찬찭찮찯찰찱찲찳찴찵찶찷참찹찺찻찼창찾찿챀챁챂챃채책챆챇챈챉챊챋챌챍챎챏챐챑챒챓챔챕챖챗챘챙챚챛챜챝챞챟챠챡챢챣챤챥챦챧챨챩챪챫챬챭챮챯챰챱챲챳챴챵챶챷챸챹챺챻챼챽챾챿첀첁첂첃첄첅첆첇첈첉첊첋첌첍첎첏첐첑첒첓첔첕첖첗처척첚첛천첝첞첟철첡첢첣첤첥첦첧첨첩첪첫첬청첮첯첰첱첲첳체첵첶첷첸첹첺첻첼첽첾첿쳀쳁쳂쳃쳄쳅쳆쳇쳈쳉쳊쳋쳌쳍쳎쳏쳐쳑쳒쳓쳔쳕쳖쳗쳘쳙쳚쳛쳜쳝쳞쳟쳠쳡쳢쳣쳤쳥쳦쳧쳨쳩쳪쳫쳬쳭쳮쳯쳰쳱쳲쳳쳴쳵쳶쳷쳸쳹쳺쳻쳼쳽쳾쳿촀촁촂촃촄촅촆촇초촉촊촋촌촍촎촏촐촑촒촓촔촕촖촗촘촙촚촛촜총촞촟촠촡촢촣촤촥촦촧촨

[과제연구] 최종 TTS 코드

촩촪촫촬촭촮촯촰촱촲촳촴촵촶촷촸촹촺촻촼촽촾촿쵀쵁쵂쵃쵄쵅쵆쵇쵈쵉쵊쵋쵌쵍쵎쵏쵐쵑쵒쵓쵔쵕쵖쵗쵘쵙쵚쵛최쵝쵞쵟쵠쵡쵢쵣쵤쵥쵦쵧쵨쵩쵪쵫쵬쵭쵮쵯쵰쵱쵲쵳쵴쵵쵶쵷쵸쵹쵺쵻쵼쵽쵾쵿춀춁춂춃춄춅춆춇춈춉춊춋춌춍춎춏춐춑춒춓추축춖춗춘춙춚춛출춝춞춟춠춡춢춣춤춥춦춧춨충춪춫춬춭춮춯춰춱춲춳춴춵춶춷춸춹춺춻춼춽춾춿췀췁췂췃췄췅췆췇췈췉췊췋췌췍췎췏췐췑췒췓췔췕췖췗췘췙췚췛췜췝췞췟췠췡췢췣췤췥췦췧취췩췪췫췬췭췮췯췰췱췲췳췴췵췶췷췸췹췺췻췼췽췾췿츀츁츂츃츄츅츆츇츈츉츊츋츌츍츎츏츐츑츒츓츔츕츖츗츘츙츚츛츜츝츞츟츠측츢츣츤츥츦츧츨츩츪츫츬츭츮츯츰츱츲츳츴층츶츷츸츹츺츻츼츽츾츿칀칁칂칃칄칅칆칇칈칉칊칋칌칍칎칏칐칑칒칓칔칕칖칗치칙칚칛친칝칞칟칠칡칢칣칤칥칦칧침칩칪칫칬칭칮칯칰칱칲칳카칵칶칷칸칹칺칻칼칽칾칿캀캁캂캃캄캅캆캇캈캉캊캋캌캍캎캏캐캑캒캓캔캕캖캗캘캙캚캛캜캝캞캟캠캡캢캣캤캥캦캧캨캩캪캫캬캭캮캯캰캱캲캳캴캵캶캷캸캹캺캻캼캽캾캿컀컁컂컃컄컅컆컇컈컉컊컋컌컍컎컏컐컑컒컓컔컕컖컗컘컙컚컛컜컝컞컟컠컡컢컣커컥컦컧컨컩컪컫컬컭컮컯컰컱컲컳컴컵컶컷컸컹컺컻컼컽컾컿케켁켂켃켄켅켆켇켈켉켊켋켌켍켎켏켐켑켒켓켔켕켖켗켘켙켚켛켜켝켞켟켠켡켢켣켤켥켦켧켨켩켪켫켬켭켮켯켰켱켲켳켴켵켶켷켸켹켺켻켼켽켾켿콀콁콂콃콄콅콆콇콈콉콊콋콌콍콎콏콐콑콒콓코콕콖콗콘콙콚콛콜콝콞콟콠콡콢콣콤콥콦콧콨콩콪콫콬콭콮콯콰콱콲콳콴콵콶콷콸콹콺콻콼콽콾콿쾀쾁쾂쾃쾄쾅쾆쾇쾈쾉쾊쾋쾌쾍쾎쾏쾐쾑쾒쾓쾔쾕쾖쾗쾘쾙쾚쾛쾜쾝쾞쾟쾠쾡쾢쾣쾤쾥쾦쾧쾨쾩쾪쾫쾬쾭쾮쾯쾰쾱쾲쾳쾴쾵쾶쾷쾸쾹쾺쾻쾼쾽쾾쾿쿀쿁쿂쿃쿄쿅쿆쿇쿈쿉쿊쿋쿌쿍쿎쿏쿐쿑쿒쿓쿔쿕쿖쿗쿘쿙쿚쿛쿜쿝쿞쿟쿠쿡쿢쿣쿤쿥쿦쿧쿨쿩쿪쿫쿬쿭쿮쿯쿰쿱쿲쿳쿴쿵쿶쿷쿸쿹쿺쿻쿼쿽쿾쿿퀀퀁퀂퀃퀄퀅퀆퀇퀈퀉퀊퀋퀌퀍퀎퀏퀐퀑퀒퀓퀔퀕퀖퀗퀘퀙퀚퀛퀜퀝퀞퀟퀠퀡퀢퀣퀤퀥퀦퀧퀨퀩퀪퀫퀬퀭퀮퀯퀰퀱퀲퀳퀴퀵퀶퀷퀸퀹퀺퀻퀼퀽퀾퀿큀큁큂큃큄큅큆큇큈큉큊큋큌큍큎큏큐큑큒큓큔큕큖큗큘큙큚큛큜큝큞큟큠큡큢큣큤큥큦큧큨큩큪큫크큭큮큯큰큱큲큳클큵큶큷큸큹큺큻큼큽큾큿킀킁킂킃킄킅킆킇킈킉킊킋킌킍킎킏킐킑킒킓킔킕킖킗킘킙킚킛킜킝킞킟킠킡킢킣키킥킦킧킨킩킪킫킬킭킮킯킰킱킲킳킴킵킶킷킸킹킺킻킼킽킾킿타탁탂탃탄탅탆탇탈탉탊탋탌탍탎탏탐탑탒탓탔탕탖탗탘탙탚탛태택탞탟탠탡탢탣탤탥탦탧탨탩탪탫탬탭탮탯탰탱탲탳탴탵탶탷탸탹탺탻탼탽탾탿턀턁턂턃턄턅턆턇턈턉턊턋턌턍턎턏턐턑턒턓턔턕턖턗턘턙턚턛턜턝턞턟턠턡턢턣턤턥턦턧턨턩턪턫턬턭턮턯터턱턲턳턴턵턶턷털턹턺턻턼턽턾턿텀텁텂텃텄텅텆텇텈텉텊텋테텍텎텏텐텑텒텓텔텕텖텗텘텙텚텛템텝텞텟텠텡텢텣텤텥텦텧텨텩텪텫텬텭텮텯텰텱텲텳텴텵텶텷텸텹텺텻텼텽텾텿톀톁톂톃톄톅톆톇톈톉톊톋톌톍톎톏톐톑톒톓톔톕톖톗톘톙톚톛톜톝톞톟토톡톢톣톤톥톦톧톨톩톪톫톬톭톮톯톰톱톲톳톴통톶톷톸톹톺톻톼톽톾톿퇀퇁퇂퇃퇄퇅퇆퇇퇈퇉퇊퇋퇌퇍퇎퇏퇐퇑퇒퇓퇔퇕퇖퇗퇘퇙퇚퇛퇜퇝퇞퇟퇠퇡퇢퇣퇤퇥퇦퇧퇨퇩퇪퇫퇬퇭퇮퇯퇰퇱퇲퇳퇴퇵퇶퇷퇸퇹퇺퇻퇼퇽퇾퇿툀툁툂툃툄툅툆툇툈툉툊툋툌툍툎툏툐툑툒툓툔툕툖툗툘툙툚툛툜툝툞툟툠툡툢툣툤툥툦툧툨툩툪툫투툭툮툯툰툱툲툳툴툵툶툷툸툹툺툻툼툽툾툿퉀퉁퉂퉃퉄퉅퉆퉇퉈퉉퉊퉋퉌퉍퉎퉏퉐퉑퉒퉓퉔퉕퉖퉗퉘퉙퉚퉛퉜퉝퉞퉟퉠퉡퉢퉣퉤퉥퉦퉧퉨퉩퉪퉫퉬퉭퉮퉯퉰퉱퉲퉳퉴퉵퉶퉷퉸퉹퉺퉻퉼퉽퉾퉿튀튁튂튃튄튅튆튇튈튉튊튋튌튍튎튏튐튑튒튓튔튕튖튗튘튙튚튛튜튝튞튟튠튡튢튣튤튥튦튧튨튩튪튫튬튭튮튯튰튱튲튳튴튵튶튷트특튺튻튼튽튾튿틀틁틂틃틄틅틆틇틈틉틊틋틌틍틎틏틐틑틒틓틔틕틖틗틘틙틚틛틜틝틞틟틠틡틢틣틤틥틦틧틨틩틪틫틬틭틮틯티틱틲틳틴틵틶틷틸틹틺틻틼틽틾틿팀팁팂팃팄팅팆팇팈팉팊팋파팍팎팏판팑팒팓팔팕팖팗팘팙팚팛팜팝팞팟팠팡팢팣팤팥팦팧패팩팪팫팬팭팮팯팰팱팲팳팴팵팶팷팸팹팺팻팼팽팾팿퍀퍁퍂퍃퍄퍅퍆퍇퍈퍉퍊퍋퍌퍍퍎퍏퍐퍑퍒퍓퍔퍕퍖퍗퍘퍙퍚퍛퍜퍝퍞퍟퍠퍡퍢퍣퍤퍥퍦퍧퍨퍩퍪퍫퍬퍭퍮퍯퍰퍱퍲퍳퍴퍵퍶퍷퍸퍹퍺퍻퍼퍽퍾퍿펀펁펂펃펄펅펆펇펈펉펊펋펌펍펎펏펐펑펒펓펔펕펖펗페펙펚펛펜펝펞펟펠펡펢펣펤펥펦펧펨펩펪펫펬펭펮펯펰펱펲펳펴펵펶펷편펹펺펻펼펽펾펿폀폁폂폃폄폅폆폇폈평폊폋폌폍폎폏폐폑폒폓폔폕폖폗폘폙폚폛폜폝폞폟폠폡폢폣폤폥폦폧폨폩폪폫포폭폮폯폰폱폲폳폴폵폶폷폸폹폺폻폼폽폾폿퐀퐁퐂퐃퐄퐅퐆퐇퐈퐉퐊퐋퐌퐍퐎퐏퐐퐑퐒퐓퐔퐕퐖퐗퐘퐙퐚퐛퐜퐝퐞퐟퐠퐡퐢퐣퐤퐥퐦퐧퐨퐩퐪퐫퐬퐭퐮퐯퐰퐱퐲퐳퐴퐵퐶퐷퐸퐹퐺퐻퐼퐽퐾퐿푀푁푂푃푄푅푆푇푈푉푊푋푌푍푎푏푐푑푒푓푔푕푖푗푘푙푚푛표푝푞푟푠푡푢푣푤푥푦푧푨푩푪푫푬푭푮푯푰푱푲푳푴푵푶푷푸푹푺푻푼푽푾푿풀풁풂풃풄풅풆풇품풉풊풋풌풍풎풏풐풑풒풓풔풕풖풗풘풙풚풛풜풝풞풟풠풡풢풣풤풥풦풧풨풩풪풫풬풭풮풯풰풱풲풳풴풵풶풷풸풹풺풻풼풽풾풿퓀퓁퓂퓃퓄퓅퓆퓇퓈퓉퓊퓋퓌퓍퓎퓏퓐퓑퓒퓓퓔퓕퓖퓗퓘퓙퓚퓛퓜퓝퓞퓟퓠퓡퓢퓣퓤퓥퓦퓧퓨퓩퓪퓫퓬퓭퓮퓯퓰퓱퓲퓳퓴퓵퓶퓷퓸퓹퓺퓻퓼퓽퓾퓿픀픁픂픃프픅픆픇픈픉픊픋플픍픎픏픐픑픒픓픔

[과제연구] 최종 TTS 코드

```
픕픖픗픘픙픚픛픜픝픞픟피픽픾픿핀핁핂핃필핅핆핇핈핉핊핋핌핍핎핏핐핑핒핓핔핕핖핗하학핚핛한핝핞핟할핡핢핣핤핥핦핧함합핪핫핬항핮핯핰핱핲핳해핵핶핷핸핹핺핻핼핽핾핿햀햁햂햃햄햅햆햇했행햊햋햌햍햎햏햐햑햒햓햔햕햖햗햘햙햚햛햜햝햞햟햠햡햢햣햤향햦햧햨햩햪햫햬햭햮햯햰햱햲햳햴햵햶햷햸햹햺햻햼햽햾햿헀헁헂헃헄헅헆헇허헉헊헋헌헍헎헏헐헑헒헓헔헕헖헗험헙헚헛헜헝헞헟헠헡헢헣헤헥헦헧헨헩헪헫헬헭헮헯헰헱헲헳헴헵헶헷헸헹헺헻헼헽헾헿혀혁혂혃현혅혆혇혈혉혊혋혌혍혎혏혐협혒혓혔형혖혗혘혙혚혛혜혝혞혟혠혡혢혣혤혥혦혧혨혩혪혫혬혭혮혯혰혱혲혳혴혵혶혷호혹혺혻혼혽혾혿홀홁홂홃홄홅홆홇홈홉홊홋홌홍홎홏홐홑홒홓화확홖홗환홙홚홛활홝홞홟홠홡홢홣홤홥홦홧홨황홪홫홬홭홮홯홰홱홲홳홴홵홶홷홸홹홺홻홼홽홾홿횀횁횂횃횄횅횆횇횈횉횊횋회획횎횏횐횑횒횓횔횕횖횗횘횙횚횛횜횝횞횟횠횡횢횣횤횥횦횧효횩횪횫횬횭횮횯횰횱횲횳횴횵횶횷횸횹횺횻횼횽횾횿훀훁훂훃후훅훆훇훈훉훊훋훌훍훎훏훐훑훒훓훔훕훖훗훘훙훚훛훜훝훞훟훠훡훢훣훤훥훦훧훨훩훪훫훬훭훮훯훰훱훲훳훴훵훶훷훸훹훺훻훼훽훾훿휀휁휂휃휄휅휆휇휈휉휊휋휌휍휎휏휐휑휒휓휔휕휖휗휘휙휚휛휜휝휞휟휠휡휢휣휤휥휦휧휨휩휪휫휬휭휮휯휰휱휲휳휴휵휶휷휸휹휺휻휼휽휾휿흀흁흂흃흄흅흆흇흈흉흊흋흌흍흎흏흐흑흒흓흔흕흖흗흘흙흚흛흜흝흞흟흠흡흢흣흤흥흦흧흨흩흪흫희흭흮흯흰흱흲흳흴흵흶흷흸흹흺흻흼흽흾흿힀힁힂힃힄힅힆힇히힉힊힋힌힍힎힏힐힑힒힓힔힕힖힗힘힙힚힛힜힝힞힟힠힡힢힣'

previous_letters_number = 0
Consonant_number = [2, 1, 2, 1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 1]
Consonants_number = [1, 3, 3, 1, 8, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1]
Consonant = [['ㄱ', 'ㄲ'], ['ㄴ'], ['ㄷ', 'ㄸ'], ['ㄹ'], ['ㅁ'], ['ㅂ', 'ㅃ'], ['ㅅ', 'ㅆ'], ['ㅇ'], ['ㅈ', 'ㅉ'], ['ㅊ'], ['ㅋ'],
             ['ㅌ'], ['ㅍ'], ['ㅎ']]

Consonants = [[' '], ['ㄱ', 'ㄲ', 'ㄳ'], ['ㄴ', 'ㄵ', 'ㄶ'], ['ㄷ'], ['ㄹ', 'ㄺ', 'ㄻ', 'ㄼ', 'ㄽ', 'ㄾ', 'ㄿ', 'ㅀ'], ['ㅁ'],
              ['ㅂ', 'ㅄ'], ['ㅅ', 'ㅆ'], ['ㅇ'], ['ㅈ', 'ㅉ'], ['ㅊ'], ['ㅋ'], ['ㅌ'], ['ㅍ'], ['ㅎ']]
Vowel = ['ㅏ', 'ㅐ', 'ㅑ', 'ㅒ', 'ㅓ', 'ㅔ', 'ㅕ', 'ㅖ', 'ㅗ', 'ㅘ', 'ㅙ', 'ㅚ', 'ㅛ', 'ㅜ', 'ㅝ', 'ㅞ', 'ㅟ', 'ㅠ', 'ㅡ', 'ㅢ', 'ㅣ']

Z = ['ㄱ', 'ㄴ', 'ㄷ', 'ㄹ', 'ㅁ', 'ㅂ', 'ㅅ', 'ㅇ', 'ㅈ', 'ㅊ', 'ㅋ', 'ㅌ', 'ㅍ', 'ㅎ']
# 자동으로 글자 단위로 끊는 알고리즘
cntind = 0
char_result = []
flag = False
while (True):
    # print(cntind)
    if flag: break
    a = ''
    b = ''
    c = ''
    cnt = 0
    # 종료조건 설정
    if cntind == len(han_result): break
    if cntind == len(han_result) - 1:
```

```
            if han_result[cntind] in Z:
                a = han_result[cntind - 2]
                b = han_result[cntind - 1]
                c = han_result[cntind]
            elif han_result[cntind] in Vowel:
                a = han_result[cntind - 1]
                b = han_result[cntind]
                c = ''
        # 특수한 경우는 따로 처리
        elif len(han_result) == 2:
            a = han_result[0]
            b = han_result[1]
            c = ''
            flag = True
        elif len(han_result) == 3:
            a = han_result[0]
            b = han_result[1]
            c = han_result[2]
            flag = True
        # 1. 모음-자음-자음 순서일 때 : 자음에서 바꾸기
        elif cntind >= 1 and han_result[cntind - 1] in Vowel and han_result[cntind] in Z and han_result[cntind + 1] in
Z:
            a = han_result[cntind - 2]
            b = han_result[cntind - 1]
            c = han_result[cntind]
        # 2. 모음-자음-모음순일때 : 모음에서 바꾸기
        elif cntind >= 1 and han_result[cntind - 1] in Vowel and han_result[cntind] in Z and han_result[
            cntind + 1] in Vowel:
            a = han_result[cntind - 2]
            b = han_result[cntind - 1]
            c = ''
            # cntind = cntind + 1

        # 합성하기.
        if a != '' and b != '':
            letter = [a, b, c]
            # 입력한 글자의 초성보다 앞에 있는 초성 수 세기
            # 한 초성당 588개의 글자 존재함. 따라서 그 초성 수에 588을 곱해 cnt에 더한다.
            for i in range(len(Consonant)):
```

[과제연구] 최종 TTS 코드

```
            if letter[0] in Consonant[i]:

                for x in range(0, i):
                    cnt += Consonant_number[x] * 588

                for j in range(len(Consonant[i])):
                    if letter[0] == Consonant[i][j]:
                        cnt += j * 588

        # 입력한 글자의 중성보다 앞에 있는 중성 수 세기
        # 초성이 결정된 상태에서 한 중성 당 28개의 글자 존재함. 따라서 그 중성 수에  28을 곱해 cnt에 더한다.
        for k in range(len(Vowel)):
            if letter[1] == Vowel[k]:
                cnt += (k) * 28
        # 입력한 글자의 종성보다 앞에 있는 종성 수 세기
        # 이 수까지 cnt에 더하면, 입력한 글자보다 앞에 있는 모든 글자의 수가 cnt가 된다.
        for l in range(0, len(Consonants)):
            if letter[2] in Consonants[l]:
                for m in range(0, l):
                    cnt += Consonants_number[m]
                for n in range(0, len(Consonants[l])):
                    if letter[2] == Consonants[l][n]:
                        cnt += n
        # 배열에 추가하기
        char_result.append(M[int(cnt)])
    cntind = cntind + 1
# print(char_result)

# 하나의 단어로 합치기
total_char = ''
for i in range(len(char_result)):
    total_char = total_char + char_result[i]
print(total_char)

# (최종)TTS변환
tts_file = gTTS(text=total_char, lang='ko')
tts_file.save("FinalTTSFile.mp3")
print('저장 완료!')
```

[과제연구] 최종 TTS 코드