import tensorflow as tf import os import cv2 import numpy as np import re from sklearn.preprocessing import LabelEncoder from sklearn.preprocessing import OneHotEncoder from numpy import array from gtts import gTTS ##cnn이미지 변환은 '모두의 머신러닝'을 강의하신 김성훈 교수님의 코드를 참고함 (https://goo.gl/CjHapo) ###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.
       imq_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(imq)])
       #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)
###테스트 데이터 가공
#학습 데이터와 같은 원리 (설명 생략)
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 = "
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)
#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
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) #과적함 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))
#평균 엔트로피 함수
optimizer = tf.train.AdamOptimizer(learning_rate).minimize(cost)
#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
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
        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))
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 초성을 출력한다.
#현재 index 0 = ㄱ 1 = ㄹ 2 = ㅍ 3 = ㅎ
char = {0:'¬', 1:'¬', 2:'¬', 3:'¬', 4:'¬', 5:'¬', 6:'¬', 7:'¬', 8:'¬', 9:'¬', 10:'¬',
        11: '=', 12: '=', 13: 'o', 14: '\', 15: '\', 16: '\', 17: '\', 18: '\',
      19: 'ヰ', 20: 'ㅜ', 21: 'ㅠ', 22: 'ㅡ', 23: '┃', 24: '┃', 25: '┃', 26: '┃', 27: '┃', 28: 'Д',
        29: 'ــ', 30: 'ד', 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 = i
   print(char[ind])
###TTSEIIOIEI
#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)
```

 ⋈ = '가각갂갃간갅갆갇갈칽갊갋갌갍갎갏감갑값갓갔강갖갗갘같갚갛개객갞갟갠갡갢갣갤갥갦갧갨갩갪갫갬갭갮갯갰 갱갲갳갴갵갶갷갸갹갺갻갼갽갾갿캴캵걂걃걄걅걆걇걈걉걊걋걌걍걎걏컄걑걒걓걔걕걖걗걘걙걚걛걜걝걞걟걠걡걢걣걤 걥쟶걧걨걩걪걫걬걭걮걯거걱걲걳건걵걶겉걸걹걺걻걼걽걾걿검겁겂것겄겅겆겇겈겉겊겋게겍겎겏겐겑겒겓겔겕겖겗겘 겙겚겛겜겝겞겟겠겡겢켗겤겥겦겧겨격쳒겫견겭겮겯켤겱겲겳겴겵겶겷겸겹졊겻겼경겾겿곀꼍곂곃계곅곆곇곈곉곊곋꼘 졝곎곏곐곑곒곓곔곕곖곗곘곙곚곛곜곝곞곟고굑곢곣곤곥곦곧골곩곪곫곬괉곮곯곰곱곲곳곴공곶곷굨곹곺곻과곽곾곿만 괁괂괃괄괅괆괇괈괉괊괋괌괍괎괏괐광괒괓괔괕괖괗괘괙괚괛먠괝괞괟괠괡괢괣괤괥괦괧괨괩괪괫괬괭괮괯괰괱괲괳괴 뫽괶뫿괸괹괺괻괼괽괾괿굀굁굂굃굄굅굆굇굈굉굊굋굌굍굎굏교굑굒굓굔굕굖굗굘굙굚굛굜굝굞굟굠굡굢굣굤굥굦굧굨 굩굪굫구국굮굯군굱굲굳굴쿩굶굷굸굹굺굻굼굽굾굿궀궁궂궃궄궅궆궇궈궉궊궋권궍궎궏궐궑궒궓궔궕궖궗궘궙궚궛궜 궝궞궟궠궡궢궣궤궥궦궧궨궩궪궫궬궭궮궯궰궱궲궳궴궵궶궷궸궹궺궻궼궽궾궿귀귁귂귃귄귅귆귇귈귉귊귋귌귍귎귏귐 귑귒귓귔귕귖귗귘귙귚귛규귝귞귟굔귡귢疋뀰큙귦귧귨귩귪귫귬귭귮귯귟귬귲귳귴귵귶귷그극귺귻근귽귾귿글긁긂긃긄

긅긆긇금급긊긋긌긍긎궂긐긑긒긓긔긕긖긗긘긙긚긛긜긝긞긟긠긡긢긣긤긥긦긧긨긩긪긫킠긭긮긯기긱긲긳긴긵긶긷길 긹긺긻긼긽긾긿김깁깂깃깄깅깆깇깈깉깊깋까깍깎깏깐깑깒깓깔깕깖깗깘깙깚깛깜깝깞깟깠깡깢깣깤깥깦깧깨깩깪깫짼 깭깮깯깰깱깲깳깴깵깶깷깸깹깺깻깼깽깾깿꺀꺁꺂꺃꺄꺅꺆꺇꺈꺉꺊꺋꺌꺍꺎꺏꺐꺑꺒꺓꺔꺕꺖꺗꺘꺙꺚꺛꺜꺝꺞꺟꺠 꺡꺢꺣꺤꺥꺦꺧꺨꺩꺪꺫꺬꺭꺮꺯꺰꺱꺲꺳꺴꺵꺶꺷꺸꺹꺺꺻꺼꺽꺾꺿껀껁껂껃껄껅껆껇껈껉껊껋껌껍껎껏껐껑껒껓껔 껕껖껗께껙껚껛껜껝껞껟껠껡껢껣껤껥껦껧껨껩껪껫껬껭껮껯껰껱껲껳껴껵껶껷껸껹껺껻껼껽껾껿꼀꼁꼂꼃꼄꼅꼆꼇꼈 꼉꼊꼋꼌꼍꼎꼏꼐꼑꼒꼓꼔꼕꼖꼗꼘꼙꼚꼛꼜꼝꼞꼟꼠꼡꼢꼣꼤꼥꼦꼧꼨꼩꼪꼫꼬꼭꼮꼯꼰꼱꾢꼳꼴꼵꼶꼷꼸꼹꼺꼻꼼 꼽꼾꼿꽀꽁꽂꽃꽄꽅꽆꽃꽈꽉꽊꽋꽌꽍꽎꽏꽐꽑꽒꽓꽔꽕꽖꽗꽘꽙꽚꽛꽜꽝꽞꽟꽠꽡꽢꽣꽤꽥꽦꽧꽨꽩꽪꽫꽼꽭꽮꽰 꽱꽲꽳꽴꽵꽶꽷꽸꽹꽺꽻꽼꽽꽾꽿꾀꾁꾂꾃꾄꾅꾆꾇꾈꾉꾊꾋꾌꾍꾎꾏꾐꾑꾒꾓꾔꾕꾗꾗꾘꾙꾚꾛꾜꾝꾞꾟꾠꾡꾢꾣꾤 꾥꾦꾧꾨꾩꾪꾫꾬꾭꾮꾯꾰꾱꾲꾳꾴꾵꾶꾷꾸꾹꾺꾻꾼꾽꾾꾿꿀꿁꿂꿃꿄꿅꿆꿇꿈뀹꿊꿋꿌꿍꿎꿏꿐꿑꿒꿓꿔꿕꿖꿗꿘 꿙꿚꿛꿬꿝꿞꿟꿠꿡꿢꿣꿤꿥꿦뀟꿨꿩꿪꿫꿬꿭꿙꿯꿰꿱꿲쀇꿴꿵꿶쪧뀈꿹꿺꿻꿼꿽꿾꿿뀀뀁뀂뀃뀄뀅뀆꿱풱풸뀆뀆퓌 뀍뀎뀏뀐뀑뀒뀥뀔꿝뀖뀗뀘뀙뀚뀛뀜뀝뀞꾓뀠뀡뀢뀣뀤뀥뀦뀧뀨뀩뀪뀫뀬뀭뀮뀯뀰뀱뀲뀳뀴뀵뀶뀷뀸뀹뀺뀻뀼뀽뀾뀿끀 끁끂끃끄끅끆끇끈끉끊끋뀰끍끎끏끐끑끒뀷끔끕끖꼿끘끙끚꿏끜끁끞끟끠끡끢끣끤끥끦끧끨끩끪끫끬끭끮끯끰끱끲끳끴 끵끶끷끸끹끺끻끼끽끾끿낀낁낂낃낄낅낆낇낈낉낊낋낌낍낎낏낐낑낒낓낔낕낖낗나낙낚낛난낝낞낟날낡낢낣낤낥낦낧남 납낪낫났낭낮낯낰낱낲놯내낵낶낷낸낹낺낻낼낽낾낿냀냁냂냃냄냅냆냇냈냉냊냋냌냍냎냏냐냑냒냓냔냕냖냗냘냙냚냛냜 냝냞냟냠냡냢냣냤냥냦냧냨냩냪냫냬냭냮냯냰냱냲냳냴냵냶냷냸냹냺냻냼냽냾냇넀넁넂넃넄넅넆넇너넉넊넋넌넍넎넏널 녉녊녋녌녍녎녏뎜녑녒녓녔뎡녖녗녈녙녚녛뎨녝녞녟녠녡녢녣녤넭넮녧넰넱녪녫녬녭녮넻녰녱넺넻녤녵녶녷뎐틕퉊녃뎐 녅녆녇녈녉녊녋녌녍녎녏녂녑녒녓녔녕녖녗녘녙녚녛뎨녝녞녟녠녡녢녣녤녥녦녧녨녩녪녫녬녭녮녯녰녱녲녳녴녵녶녷노 녹녺녻논녽녾녿놀놁놂놃놄놅놆놇**놈놉놊놋놌**농놎놏놐놑높놓놔놕놖놗놘놙놚놛놜놝놞놟놠놡놢놣놤놥놦놧놨놩놪놫놬 놭놮놯놰놱놲놳놴놵놶놷놸놹놺놻놼놽놾놿뇀뇁뇂뇃뇄뇅뇆뇇뇈뇉뇊뇋뇌뇤뇎뇏뇐뇑뇒뇓뇔뇕뇖뇗뇘뇙뇚뇛뇜뇝뇞뇟뇠 뇡뇢뇣뇤뇥뇦뇧뇨뇩뇪뇫뇬뇭뇮뇯뇰뇱뇲뇳뇴뇵뇶뇷뇸뇹뇺뇻뇼뇽뇾뇿눀눁눂눃누눅눆눇눈눉눊눋눌눍눎눏눐눑눒눓눔 눕눖눗눘눙눚눛눜눝눞눟눠눡눢눣눤눥눦눹눨눩눪눫눬눭눮눯눰눱눲눳눴눵눶뒃눰눹눺눻눼눽눾눿뉀뉁뉂뉃뉄뉅뉆뉇뉈 뉉뉊뉋뉌뉍뉎뉏뉐뉑뉒뉓뉔뉕뉖뉗뉘뉙뉚뉛뒨뉝뉞뉟뉠뉡눪뉣뉤뉥뉦뉧뒴뷥뉪뉫뉬뉭뉮뉯뒼뉱뉲뒿뉴뉵뉶뉷듄뉹뉺늍뉼 뉽뉾뉿늀늁늂늃늄늅늆늇늈늉늊늋늌늍늎늏느늑늒늓는늕늖늗늘눍늚늛늜늝늞늟늠늡늢늣늤능늦늧늨늩늪늫늬늭늮늯늰 늱늲늳늴늵늶늷늸늹늺늻늼늽늾늿닀닁닂닃닄닅닆닇니닉닊닋닌**닍닎닏닐닑닒닓닔닕닖닗**님닙닚닛닜닝닞닟닠닡닢닣다 <u>닥닦닧단닪닪닫돨닭닮돫닰돭탎돯담답닶닷닸당닺돷돸탙닾돻대댁댂댃댄댆댆댇댈댉댊댋댌댍댎댏댐댑댒댓댔댕댖댗댘</u> 댙댚댛댜댝댞댟댠댡댢댣댤댥댦댧댨댩댪댫댬댭댮댯댰댱댲댳댴댵댶댷댸댹댺댻댼댽댾댿덀덁덂덃덄덅덆덇덈덉덊덋덌 덍덎덏덐덑덒덓더덕덖덗던덙덚덛덜덝덞덟덠덡덢덣덤덥덦덧덨덩덪덫덬덭덮덯데덱덲덳덴덵덶덷델덹덺덻덼덽덾덿뎀 뎁뎂뎃뎄뎅뎆뎇뎈뎉뎊뎋뎌뎍뎎뎏뎐뎑뎒뎓뎔뎕뎖뎗뎘뎙뎚뎛뎜뎝뎞뎟뎠뎡뎢뎣뎤뎥뎦뎧뎨뎩뎪뎫뎬뎭뎮뎯뎰뎱뎲뎳뎴 뎵뎶뎷뎸뎹뎺뎻뎼뎽뎾뎿돀돁돂**돃도독돆돇돈돉돊**돋돌돍돎돏돐돑돒돓돔돕돖돗돘동돚돛돜돝돞돟돠돡돢돣돤돥돦돧돨 돩돪돫돬돭돮돯돰돱돲돳돴돵돶돷돸돹돺돻돼돽돾돿됀됁됂됃됄됅됆됇됈됉됊됋됌됍됎됏됐됑됒됓됔됕됖됗되됙됚됛된 됝됞됟될됡됢됣됤됥됦됧됨뒵됪됫됬됭됮됯됰됱됲됳됴됵됶됷됸됹됺됻됼됽됾됿둀둁둂둃둄둅둆둇둈둉둊둋둌둍둎둏두 둑둒둓둔둕둖둗둘둙둚둛둜둝둞둟둠둡둢둣둤둥둧둧둨둩둪둫둬둭둮둯둰둱둲둳둴둵둶둷둸둹둺둻둼둽둾퉛뒀뒁뒂뒃뒄 뒅뒆뒇뒈뒉뒊뒋뒌뒍뒎뒏뒐뒑뒒뒓뒔뒕뒖뒗뒘뒙뒚퉷뒜뒝뒞뒟뒠뒡뒢뒣뒤뒥뒦뒧뒨뒩뒪뒫뒬뒭뒮뒯뒰뒱뒲뒳뒴튑뒶튓뒸 뒹뒺뒻뒼뒽뒾뒿듀듁듂듃듄듅듆듇듈듉듊듋듌듍듎듏듐듑듒듓듔듕듖듗듘듙듚듛드륵듞듟뜬듡듢듣들듥듦듧듨듩틆듫듬 <u>틉듮돗듰틍둦둫듵틒둫듸띜듺듻듼듽듾뒫딀딁딂딃딄딅딆딇뒴딉딊딋딌딍딎딏딐딑딒됳디딕딖딗뎐딙딚딛딜딝딞딟딠</u> 딡딢딣딤딥딦딧딨딍딪딫팈딭딮딯따딱딲딳딴딵딶땉똴딹딺딻딼딽딾딿땀땁땂땃땄뙁뙂땇땈땉땊땋떄땍땎땏땐땒땒땓땔 땕땖땗땘땙땚땛땜땝땞땟땠땡땢땣땤땥땦땧땨땩땪땫땬땭땮땯땰땱땲땳땴땵땶땷땸땹땺땻땼땽땾땿떀떁떂떃떄떅떆떇떈 떉떊떋떌떍떎떏떐떑떒떓떔떕떖땟떘떙떚떛떜떝떞떟떠떡떢떣떥떥떦떧떨떩떪떫떬떭떮떯떰떱떲떳떴떵떶떷떸떹떺떻떼 떽떾떿뗀뗁뗂뗃뗄뗅뗆뗇뗈뗉뗊뗋뗌뗍뗎뗏뗐뗑뗒뗓뗔뗕뗖뗗뗘뗙뗚뗛뎐뗝뗞멷뗠뗡뗢뗣뗤뗥뗦뗧뗨뗩뗪뗫뗬뗭뗮뗯뗰

뗱뗲뗳뗴떽뗶뗷뗸몑뗺뗻똌뗽뗾뗿똀똁똂똃똄똅똆똇똈똉똊뗓똌똍똎똋도똨똒똓똔똕똖똗똨똙똚똛똜똝똞똟똠똡똢똣뚔 똥똦똧똨똩똪똫똬똭똮똯똰똱똲똳똴똵똶똷똸똹똺똻똼똽똾똿뙀뙁뙂뙃뙄뙅뙆뙇뙈뙉뙊뙋뙌뙍뙎뙏뙐뙑뙒뙓뙔뙕뙖뙗뙘 뙙뙚뙛뙜뙝뙞뙟뙠뙡뙢뙣뙤뙥뙦뙧뙨뙩뙪뙫뙬뙭뙮뙯뙰뙱뙲뙳뙴뙵뙶뙷뙸뙹뙺뙻뙼뙽뙾뙿뚀뚁뚂뚃뚄뚅뚆뚇뚈뚉뚊뚋뚌 뚍뚎뚏뚐뚑뚒뚓뚔뚕뚖뚗뚘뚙뚚뚛뚜뚝뚞뚟뚠뚡뚢뚣뚤뚥뚦뚧뚨뚩뚪뚫뚬뚭뚮뚯뚰뚱뚳뚳뚴뚵뚶뚳뚸뚹뚺뚻뚼뚽뚾뚿뛀 **뛁뛂뛃뛄뛅뛆뛇뛈뛉뜂뛋뛌뛍뛎뛏뛐뛑뛒**뛓뛔뛕뛖뛗뛘뛙뛚뛛뛜뛝뛞뛟뛠뛡뛢뛣뛤뛥뛦뛧뛨뛩뛪뛫뗔뛭뛯顺厈뮉뛲뛳뛴 뛵뛶뛷뛸뛁뛺뛃뛄뛅뛾뛿뜀뜁뜂뜃뜄뜅뛎뜇뜈뛑뜊뜋뜌뜍뜎뜏뜐뜑뜒뜓뜔뜕뜖뜗뜘뜙뜚뜛뜜뜝뜞뜟뜠뜡뜢뜣뜤뜥뜦뜧뜨 <u> 똑뜪뜫뜬뜮뜮뜓뜔뜱뜲뜳뜴뜵뜶뜛뜸땹뜺똣뜼뜽뜢뚲띀띁띂띃뙤띅띆띇뛴띉띊띋띌띍띎띏띐띑띒띓띔띕띖띗띘띙띛띛띜</u> 띝띞띟띠띡띢띣띤띥띦띧띨띩띪띫띬띭띮띯띰띱띲띳띴띵띶띷**띸띹띺띻라락띾띿란랁랂랃랄랅랆랇랈랉랊랋띾**랔랎랏랐 랑랒랓랔랕랖랗래랙랚랛랜랝랞랟랠랡랢랣랤랥랦랧램랩랪랫랬랭랮랯랰랱랲랳랴략랶랷랸랹랺랻랼랽랾랿럀럁럂럃럄 럅럆럇럈량럊럋럌럍랶럏럐럑럒럓럔럕럖럗럨럙럚럛럜럝럞럟럠럡럢럣럤럥럦럧럨럩럪럫긤렄럮럯럱럱렅럳렄럵럶럷럸 럹럺럻럼럽럾럿렀렁렂렃렄렅렆렇궨렉렊렋렌렍렎렏렐렑렒렓렔렕렖렗렘렙렚렛렜렝렞렟렠렡렢렣겨력렦렧련렩렪렫렬 렭렮렯렰렱렲렳렴렵렶렷렸령렺렻렼렽렾렿뎨롁롂롃롄롅롆롇롈롉롊롋롌롍롎롏롐롑롒롓롔롕롖롗롘롙롚롛로록롞롟론 롡롢롣롤롥롦롧롨롩롪롫롬롭롮롯롰롱롲롳롴롵롶롳롸롹롺롻롼롽롾롿뢐뢁뢂뢃뢄뢅뢆뢇뢈뢉뢊뢋뢌뢍뢎뢏뢐뢑뢒뢓뢔 뢕뢖뢗뢘뢙뢚뢭뢬뢝뢞뢟뢠뢡뢢뢣뢤뢥뢦뢧뢨뢩뢪뢫뢬뢭뢮뢯쾨뢱뢲뢳뢴뢵뢶뢷뢸뢹뢺뢻뢼뢽뢾뢿룀룁룂룃룄룅룆룇룈 룉룊룋료룍룎룏룐룑룑룓룔룕룖룗룘룙룚룛룜룝룞룟룠룡룢룣룤묱룦룧루룩룪룫룬룭룮룯룰룱룲룳둜룵뭂룷룸툽룺룻룼 룽룿룿뤀뤁뤂뤃뤄뤅뤆뤇륁뤉뤊뤋뤜뤍뤏뤏뤒뤒뤒뤓뤔뤕뤖뤗뤒뤙뤛뤛뤅뤝뤞뤛뤠뤡뤢뤣뤤뤥뤦뤧뤨뤩뤫뤫뤬뤭뤮뤯뤰 뤱뤲뤳뤴뤵뤶뤻뤸뤹뤺뤻뤼뤽뤾뤿륀륁륂륃륔륅륆륇륈륉륊륋륌륍륎륏륐륑륒륓륔륕륖륗류룍륚륛륜륞륞륟륰륡륢륪륤 륥륦륧륨륩륪륫륬륭륮륯륰륱륲륳ㄹ륵륶륷른륹롡륻를륽륾륿룘릁릂릃름릅릆릇릈틍릊룿릌릍릎릏킈릑릒릓릔릕릖릗릘 릙릚릛릜릝릞릟릠릡릢릣릤릥릧릧릨릩릪릫리릭릮릯린릱릲릳릴릵릶릷릸릹릺릻림립릾킛맀링맂맃맄맅맆맇먀막맊맋만 맍많맏말맑맒맓맔맕맖맗맊먑맚맛맜먕먖맟맠땉맢맣매맥맦맧먠맩맪맫맬맭맮맯맰맱맲맳맴맵맶맷맸맹먲맻맼맽맾맿먀 먁먂먃먄먅먆먇먈먉먊먋먌먍먎먏먐먑먒먓먔먕먖먗먘먙먚먛먜먝먞먟먠먡먢먣먤먥먦먧먨먩먪먫먬먭먮맷먰먱먲먳먴 먵먶먷머먹먺먻먼먾먾멑덜멁멂멃멄멅멆멇멈멉멊멋멌멍멎멏멐멑멒멓메뭭멖멗멘멙멚멛멜멝멞멟멠멡멢멣멤멥멦멧멨 멩멪멫멬멭멮멯며멱멲멳면멵멶멷멸멹멺멻멼멽멾멿몀몁몂몃몄명몆몇몈몉몊몋몌몍몎몏몐몑몒몓몔몕몖몗몘몙몚몛몜 몝몞몟몠몡몢몣몤몥몦몧모목몪몫몬몭몮몯몰몱몲몳몴몵몶몷몸몹몺못몼몽몾몿뫀뫁뫂뫃뫄뫅뫆뫇뫈뫉뫊뫋뫌뫍뫎뫏뫐 뫑뫒뫓뫔뫕뫖뫗뫘뫙뫚뫛뫜뫝뫞뫟뫠뫡뫢왟뫤뫥왢뫧뫨뫩뫪뫫뫬뫭뫮뫯뫡뫱뫲뫳뫴뫵뫶뫷뫸뫹뫺뫻뫼뫽뫾뫿묀묁묂묃묄 묅묆묇묈묉묊묋묌묍묎묏묐묑묒묓묔묕묖묗됴묙묚묛묜묝묞묟묠묡묢묣묤묥묦묧묨묩묪묫묬묭묮묯묰묱묲묳무묵묷묷문 묹묺묻물묽묾묿뭀뭁뭂뭃뭄뭅뭆뭇뭈뭉뭋뭋뭌뭍뭎뭏뭐뭑뭒뭓뭰뭕뭖뭗뭴뭙뭚뭛뭜뭝뭞뭟뭠뭽뭢뭣뭤뭥뭦뭧뭨뭩뭪뭫뭬 뭭뭮뭯뭰뭱뭲뭳뭴뭵뭶뭷뭸뭹뭺뭻뭼뭼뭾뭿뮀뮁뮂뮂뮄뮅뮆뮇믺뮉뮊뮋뮌뮍뮎뮏뮠뮑뮒뮓뮔뮕뮖뮗뮘뮙뮚뮛뮜뮝뮞뮟뮠 뮡뮢뮣뮤뮥뮦뮧뮨뮩뮪뮫뮬뮭뮮뮯뮰뮱뮲뮳뮴뮵뮶뮷뮸뮹뮺뮻뮼뮽뮾뮿므믁믂믃믄믅믆믇믈믉믊믋믌믍믎믏믐믑믒믓믔 믕믖믗믘믙믚믛믜믝믞믟믠믡믢믣믤믥믦믧믨믩믪믫믬믭믮믯믰믱믲믳믴믵믶믷미믹믺믻민믽믾믿밀밁밂밃밄밅밆밇밈 밉밊밋밌밍밎밓밐밑밒밓바박밖밗뱐뱑밚받봘봙봚봛밠밡밢밣뱜밥밦밧밨방밪밫밬밭밮밯뱨백밲밳뱬밵뱮밷뱈밹밺밻밼 밽밾밿뱀뱁뱂뱃뱄뱅뱆뱇뱈뱉뱊뱋뱌뱍뱎뱏뱐뱑뱒뱓뱔뱕뱖뱗뱘뱙뱚뱛뱜뱝뱞뱟뱠뱡뱢뱣뱤뱥뱦뱧뱨뱩뱪뱫뱬뱭뱮뱯뱰 뱱뱲뱳뱴뱵뱶뱷뱸뱹뱺뱻뱼뱽뱾뱿벀벁벂벃버벅벆벇번벉벊벋벌벍벎벏벐벑벒벓텀법벖벗벘벙벚벛벜벝벞벟베벡벢벣벤 벥벦벧벨벩벪벫벬벭벮벯벰벱벲벳벴벵벶벷벸벹벺벻벼벽벾벿변볁볂볃별볅볆볇볈볉볊볋볌볍볎볏볐병볒볓볔볕볖볗볘 볙볚볛볜볝볞볟볠볡볢볣볤볥볦볧볚볩볪볫볬볭볮볯볰볱볲볳노복볶볷본볹볺볻볼볽볾볿봀봁봂봃볽봅봆봇봈봉봊봋봌 봍봎봏놔봑봒봓봔봕봖봗봘봙봚봛봜봝봞봟봠봡봢뵛봤봥봦봧봘봩봪봧봬봭봮봯봰봱봲봳뵄봵봶봷봸봹봺봻봼봽봾봿뵀 뵁뵂뵃뵄뵅뵆뵇뵈뵉뵊뵋뵌뵍뵎뵏뵐뵑뵒뵓뵔뵕뵖뵗뵘뵙뵚뵛뵜뵝뵞뵟뵠뵡뵢뵣뵤뵥뵦뵧뵨뵩뵪뵫뵬뵭뵮뵯뵰뵱뵲뵳뵴 뵵뵶뵷뵸뵹뵺뵻뵼뵽뵾뵿부북붂붃분붅붆붇불붉붊붋붌붍붎붏붐붑붒붓붔붕붗붗붘붙붚붛붜붝붞붟붠붡붢붣붤붥붦붧붨 붩붪붫붬붬붮뷧붰붱붲붳붴붵붶붷붸붹붺붻붼붽붾붿뷐뷁뷂뷃쀐뷅뷆뷇붹뷉쀖붻쀘뷍뷎뷏뷐뷑뷒뷓빆뷕뷖扎뷘뷙뛶뷛뷬

뷝뷞뷟뷠뷡븶뷡뷥뷥뵚붻뷌뷩뷫뷫뷬뷭뷫뷫뵤븈뷲븄뷴뷶뷶븉븈뷹뷺뷻뷺뷽뷺뷿븀븀븂븃븄븅븇븋븈븉븊븋브븍븎븏뵨 븑븒븓블븕븖븗븘븙븚븛븜븝븞븟븠븡븢붖븤븥븦븧븨븩븪븫븬븭븮븯븰븱븲븳븴븵븶븷븸븹븺븻븼븽븾븿빀빁빂빃비 빅빆빇빈빉빊빝빌빍빎빏빐빑빒빓빔빕빖빗빘빙빚빛빜빝빞빟뺘빡빢빣빤빥빦빧빨빩빪빫빬빭빮빯빰빱빲빳**빴**빵빶빷빸 빹빺빻빼빽빾빿뺀뺁뺂뺃뺄뺅뺆뺇뺈뺉뺊뺋뺌뺍뺎뺏뺐뺑뺒뺓뺔뺕뺖뺗뺘뺙뺚뺛뺜뺝뺞뺟뺠뺡뺢뺣뺤뺥뺦뺧뺨뺩뺪뺫뺬 뺭뺮뺯뺰뺱뺲뺳뺴뺵뺶뺷뺸뺹뺺뺻뺼뺽뺾뺿뻀뻁뻂뻃뻄뻅뻆뻇뻈뻉뻊뻋뻌뻍뻎뻏뻐뻑뻒뻓뻔뻕뻖뻗뻘뻙뻚뻛뻜뻝뻞뻟뻠 뻡뻢뻣뻤뻥뻦뻧뻨뻩뻪뻫뻬뻭뻮뻯뻰뻱뻲뻳뻴뻵뻶뻷뻸뻹뻺뻻뻼뻽뻾뻿뼀뼁뼂뼃뼄뼅뼆뼇뼈뼉뼊뼋뼌뱭뼎뼏뼐뼑뼒뼓뼔 뼕뼖뼗뼘뼙뼚뼛뼜뼝뼞뼟뼠뼡뼢뼣뼤뼥뼦뼧뼨뼩뼪뼫뼬뼭뼮뼯뼰뼱뼲뼳뼴뼵뼶뼷뼸뼹뼺뼻뼼뼽뼾뼻뽀뽁뽂뽃뽄뽅뽆뽇뽈 뽉뽊뽋뽌뽍뽎뽏뽁뿁뽒뽓뽔뽕뽖뽗뽈뽙뽚뽛뽜뽝뽞뽟뽠뽡뽢뽣뽤뽥뽧뽨뽩뽪뽫뽬뽭뽮뽯뽰뽱뽲뽳뽤뽵뽶뽷뽸뽹뽺뽻뽼 뽽뽾뽿뾀뾁뾂뾃뾄뾅뾆뾇뾈뾉뾊뾋뾌뾍뾎뾏뾐뾑뾒뾓뾔뾕뾖뾗뾘뾙뾚뾜뾜뾝뾞뾟뾠뾡뾢뾣뾤뾥뾦뾧뾨뾩뾪뾫뾬뾭뾮뾯뾰 뾱뾲뾳뾴뾵뾶뿊뿈뾹뾺뾻뾼뾽뾾뾿뿀뿁뿂뿃뿄뿅뿆뿇뾱뿉뿊뿋뿌뿍뿎뿠뿑뿑뿓뿔뿕뿗뿘뿙뿚뿛뿚뿚뿡뿡뿡쁫쁫쁫뿢뿣뿤 뿥뿦뿧뿨뿩뿪뿫뿬뿭뿮뿯뿰뿱뿲뿳뿴뿵뿶뿷뿸뿹뿺쀳뿼뿽뿾뿿쀀쀁쀂쀃쀄쀅쀆쀇쀈쀉쀊쀌쀍쀎쀏쀐쀑쀒쀓쀔쀕쀖쀗쀘 쀙쀚쀛쀜쀝쀞쀟쀠쀡쀢쀣쀤쀥쀦쀧쀨쀩쀪쀫쀬쀭쀮쀯쀰쀱쀲쀳쀴쀵쀶쀷쀸쀹쀺쀻쀼쀽쀾쀿쁀쁁쁂쁃쁄쁅쁆쁇쁈쁉쁊쁋쁌 쁍쁎쁏쁐쁑쁒쁓쁄쁕쁖쁗쁘쁙쁚쁛뽄쁝쁞쁟쁠쁡쁢쁣쁤쁥쁦쁧뿀뿁쁪쁫쁬쁭쁮뿢쁰쁱쁲쁳쁴쁵쁶쁷쁸쁹쁺쁻쁼쁽쁾쁿삜 삁삂삃삄삅삆삇삈삉삊삏삌삍삎삏삐삑삒삓삔삕삖삗삘삙삚삛삜삝삞삟삠삡삢삣삤삥삦삧삨삩삪삫사삭삮삯산삱삲삳살 **삵앎삷삸삹삺삻삼삽삾삿샀상샂샃샄샅샆샇새색샊샋쌘샍샎샏**샐샑샒샓샔샕샖샗샘샙샚샛샜생샞샟샠샡샢샣샤샥샦샧샨 샩샪샫샬샭샮샯샰샱샲샳샦샵샶샷샸샹샺샻샼샽샾샿쟤섁섂섃섄섅섆섇섈섉섊섋섌섍섎섏섂섑섒섓섔섕섖섗섘섙섚섛서 석섞섟선섡섢셛설셝셞셟셠섩셢섫셤셥섮셧셨셩셪섳섴섵섶셯세셱셲셳셴셵셶셷셸셹셺셻셄셅셾셇솀솁솂솃셌솅솆솇솈 셑셒셓셔셕셖셗션셙셚셛셜셝셞셟셠셡셢셣셤셥셦셧셨셩셪셫셬셭셮셯졔셱셲셳셴셵셶셷셸셹셺셻셼셽셾셿솀솁솂솃솄 솅솆솇솈솉솊솋소속솎솏손솑솒솓솔솕솖솗솘솙솚솛솜솝솞솟솠송솢솣솤솥솦솧솨솩솪솫솬솭솮솯솰솱솲솳솴솵솶솷솸 솹솺솻솼솽솾솿쇀쇁쇂쇃쇄쇅쇆쇇쇈쇉쇊쇋쇌쇍쇎쇏쇐쇑쇒쇓쇔쇕쇖쇗쇘쇙쇚쇛쇜쇝쇞쇟쇠쇡쇢쇣쇤쇥쇦쇧쇨쇩쇪쇫쇬 쇭쇮쇯쇰쇱쇲쇳쇴쇵쇶쇷쇸쇧쇺쇻쇼쇽쇾쇿숀숁숂숃숄숅숆숇숈숉숊숋숌숍숎숏숐숑숒숓숔숕숖숗수숙숚숛순숝숞숟술 숡숢숣숤숥숦숧숨숩숪숫숬숭숓숯숰숱숲숯숴숵숶숷숸숹숺숼숼숽숾숿쉀쉁쉂쉃쉄쉅쉆쉇쉈쉉쉊쉋쉌줱쉎쉏쉐쉑쉒쉓쉔 쉕쉖쉗쉘쉙쉚쉛쉜쉝쉞쉟쉠쉡쉢쉣쉤쉥쉦쉧쉨쉩쉪쉫쉬쉭쉮쉯쉰쉱쉲슅쉴쉵쉶쉷쉸쉹쉺쉻쉼쉽쉾쉿슀슁슂슃슄슅슆슃슈 슉슊슋슌슍슎슏슐슑슒슓슔슕슖슗슘슙슚슛슜슝슞슟슠슡슢슣스슥슦슧슨슩슪슫슬슭슮슯슰슱슲슳슴습슶슷슸승슺슻슼 슽슾슿싀싁싂싃싄싅싆싇싈싉싊싋싌싍싎싏싐싑싒싓싔싕싖싗싘싙싚싛시식싞싟신싡싢ዾ실싥싦싧싨싩싪싫싞십싮싯싰 싱싲싳싴싵싶싷싸싹싺싻싼싽쏺싿쏼쌁쌂쌃쌄쌅쐂쐃쌈쌉쌊쌋쌌쌍쌎쌏쌐쌑쌒쐏쌔쌕쌖쌗쌘쌙쌚쌛쌜쌝썖쌟썘쌡쌢쌣쌤 쌥쌦쌧쌨쨍쌪쌫쌬쌭쌮쌯쌰쌱쌲쌳쌴쌵쌶쌷썈쌹쌺쌻쌼쌽쌾쌿썀썁썂썃썄썅썆썇썈썉썊얳썌썍썎썏썐썑썒썓썔썕썖썗썘 썙썚썛썜썝썞썟썠썡썢썣썤썥썦썧써썩썪썫썬썮썮썯썰썱썲썳썴썵썶썷쎰썹썺썻썼썽썾썿쎀쎁쎂쎃쎄쎅쎆쎇쎈쎉쏂쎋쎌 쎍쎎쎏쎐쎑쎒쎓쎔쎕쎖쎗쎘쎙쎚쎛쎜쎝쎞쎟쎠쎡쎢쎣쎤쎥쎦쎧쎨쎩쎪쎫쎬쎭쎮쎯쎰쎱쎲쎳쎴쎵쎶쎷쎸쎹쎺쎻쎼쎽쎾쎿쏀 쏁쏂쏃쏄쏅쏆쏇쏈쏉쏊쏋쏌쏍쏎쏏쏐쏑쏒쏓쏔쏕쏖쏗쏘쏙쏚쏛쏜쏝쏞쏟쏠쏡쏢쏣쏤쏥쏦쏧쏨쏩쏪쏫쏬쏭쏮쏯쏰쏱쏲쏳쏴 쏵쏶쏷쏸쏹쏺쐍쏼쏽쏾쏿쐀쐁쐂쐃쐄쐅쐆쐇쐈쐉쐊쐋쐌쐍쐎쐏쐐쐑쐒쐓쐔쐕쐕쐗쐘쐙쐚쐛쐜쐝쐞쐟쐠쐡쐢쐣쐤쐥쐦쐧쐨 쐩쐪쐫쐬쐭쐮쐯쐰쐱쐲쐳쐴쐵쐶쐷쐸쐹쐺쐻쐼쐽쐾쐿쑀쑁쑂쑃쑄쑅쑆쑇쑈쑉쑊쑋쑌쑍쑎쑏쑐쑑쑒쑓쑔쑕쑖쑗쑘쑙쑚쑛쑜 쑝쑞쑟쑠쑡쑢쑣쑤쑥쑦쑧쑨쑩쑪쑫쑬쑭쑮쑯쑰쑱쑲쑳쑴쑵쑶쑷쑸쑹쏯쑻쑼쑽쑾쑿쒀쒁쒂쒃쒄쒅쒆쒇쒈쒉쒊쒋쒌쒍쒎쒏쒐 쒑쒒쒓쒔쒕쒖쒗쒘쒙쒚쒛쒜쒝쒞쒟쒠쒡쒢쒵쒤쒥쒦쒧쒨쒩쒪쒫쒬쒭쒮쒯쒰쒱쒳쒳쒴쒵쒶쒷쒸쒹쒺쒻쒼쒽쒾쒿씤쓁쓂쓃쓄 쓅쓆쓇쓈쓉쓊쓋쓌쓍쓎쓏쓐쓑쓒쓓쓔쓕쓖쓗쓘쓙쓚쓛쓜쓝쓞쓟쓠쓡쓢쓣쓤쓥쓦쓧쓨쓩쓪쓫쓬쓭쓮쓯쓰쓱쓲쓳쓴쓵쓶쓷쓸 쓹쓺쓻쓼쓽쓾쓿쏨쑵씂씃씄씅씆쑻씈씉쏲씋씌씍씎씏쐰씑씒쐳쐴씕씖씗쐸씙씚씛씜씝씞쐿쑀씡씢씣씤씥씦씧씨씩씪씫뭰 씭씮씯씰씱씲씳씴씵씶씷씸씹씺씻씼씽씾씿앀앁앂앃아악앆앇안앉않앋알앍앎앏앐앑앒앓암얍앖앗았앙앚앛앜앝앞앟애 **액앢앣앤앥앦앧앨앩앪앫앬앭앮앯앰앱앲앳앴앵앶앷앸앹앺앻야약앾앿얀얁얂얃얄얅얆얇얈얉얊얋얌얍얎얏얐**양얒얓얔 **얕얖얗애얙얚얛얜얝얞얟얠얡얢얣얤얥얦얧얨얩얪깻얬얭얮얯얰얱얲얳어억얶얷언얹얺얻얼얽얾얿엀엁엂엃엄헙없엇었**

엉엊엋헠엍엎엏에엑엒엓옌엕엖엗엨엙엚엛엞엝엞엟엠엡엢엣엤엥엦엧엨엩엪엫여역엮엯**연엲**엲엳열엶엶엷엸엹엺엻염 엽엾엿였영옂옃옄옅옆옇예옉옊옋옌옍옎옏옐옑옒옓옔옕옖옗옘옙옚옛옜옝옞옟옠옡옢옣오옥옦옧온옩옪옫올옭옮옯옰 옱옲옳옴옵옶옷옸옹옺옻옼옽옾읗오왁왂왃똰왅왆왇왈왉왊왋왌왍왎왏왐왑왒왓뫘왕왖왗왘왙왚왛왜왝왞왟왠왡왢왣왤 왥왦왧왨왩왪왫왬왭왮왯왰왱왲왳왴왵왶왷외왹왺왻왼왽왾욀욀욁욂욃욄욅욆욇욈욉욊욋욌욍욎욏욐욑욒욓요욕욖욗욘 욙욚욛욜욝욞욟욠욢욢욣욤욥욦욧욨용욪욫욬욭욮욯우욱욲욳운욵욶욷울욹욺욻욼욽욾욿움웁웂웃웄웅웇웇웈욷웊웇워 웍웎웏원웑웒웓월웕웖웗웘웙웚웛웜웝웞웟윘웡웢웣웤웥웦웧엚웩웪웫웬웭웮웯뷀웱웲웳웴웵웶웷웸쒭웺웻웼웽웾웿윀 윁윂윃위윅윆윇윈윉윉윋윜윍윎윏윐윑윒윓윔윕윖읫윘읭윚윛윜윝윞윛유윸윢윣윤윦윦윧율윩윪윫윬윭윮윯윰윰윲윳윴 융윶윷윸윹윢윻으윽윾윿은읁읂읃을읅읆읇읈읉읊읋음읍읎읏옸응읒읓읔읕읖읗이읙읚읛읜읝읞읟읠읡읢읣읤읥읦읧읨 읩읪읫읬읭읮읯읰읱읲읳이익읶읷인읹읺읻일읽읾읿잀잁잂잃임입잆잇있잉**잊잋잌잍잎잏자작잒잓잔잕**잖잗잘잙잚잛잜 잝잞잟잠잡잢잣잤장잦잧잨잩잪잫쟤쟄잮잯잰잱잲잳잴잵잶잷잸잹잺잻잼잽잾잿쟀쟁쟂쟃쟄쟅쟆쟇쟈쟉쟊쟋쟌쟍쟎쟏쟐 쟑쟒쟓쟔쟕쟖쟗쟘쟙쟚쟛쟜쟝쟞쟟쟠쟡쟢쟣**갞쟥쟦쟧쟨쟩쟪쟫쟬쟭쟮쟯쟰쟱쟲쟳쟴쟵쟶쟷쟸쟹쟺쟻쟼쟽쟾쟿저**적젂젃전 젅젆젇절젉젊젋젌젍젎젏점덥젒젓젔정젖젗젘젙젚젛졔젝젞젟젠젡젢젣젤젥졞젧젨젩젪젫젬젭젮젯젰젱젲젳펰젵젶젷져 젹젺젻젼젽젾젿졀졁졂졃졄졅졆졇졈졉졊졋졌졍졏졏졐졑졒졓졔졕졖졗졘졙졚졛졬졝졞졟졠졡졢졣졤졥졦졧졨졩졪졫졬 졭졮졯조족졲졳존졵졶졷졸졹졺졻졼졽졾졿좀좁좂좃좄종좆좇좈좉좊좋좌좍좎좏좐좑좒좓좔좕좖좗좘좙좚좛좜좝좞좟좠 좡좢좣좤좥좦좧좨좩좪좫좬좭좮좯좰좱좲좳좴좵좶좷좸좹좺좻좼좽좾좿죀죁죂죃**즤죅죆죇죈죉죊죋죌죍죎**죏죐죑죒죓죔 죕죖죗죘죙죚죛죜죝죞죟**죠죡죢죣죤죥죦죧죸죩죪죫죬죭죮죯죰죱죲죳죴죵죶죷죸죹죺죻주**줔죾줐준줂줕줃줄줅줆줇줈 줉줊줋죾줍줎줏줐중줓줓줔줕줖줗줘줙줚줛줜쥖줞줟줰줡줢줣줤줥줦줧줨줩줪쥣줬줭줮줯줰줱줲줳줴줵줶줷줸줹줺줻줼 줽줾줿쥀쥁쥂쥃쥄쥅쥆쥇쥈쥉쥊쥋쥌쥍쥎쥏쥐쥑쥒쥓쥔쥕쥖쥗쥘쥙쥚쥛쥜쥝쥞쥟쥠쥡쥢쥣쥤쥥쥧쥧쥨쥩쥪쥫쥬쥭쥮쥯쥰 쥲쥲쥳쥴쥵쥶쥷쥸쥹쥺쥻쥼쥼쥾쥿즀쥼즃즃쥴즅즆즇즈즉즊즋즌즎즎즏즐즑즒즓즔즕즖즗즊즙즚즛즜증즞즟즠즡즢즣즤 즥즦즧즨즩즪즫즬즭즮즯즰즱즲즳즴즵즶즷즸즹즺즻즼즽즾즿지직짂짃진짅짆짇질짉짊짋짌짍짎짏짐집짒짓짔징짖짗짘 짙짚짛짜짝짞짟짠짡짢짣쫠짥짦쫣짨짩짪짫짬짭짮짯짰**짱짲짳짴짵짶짷쩨짹쨲짻짼짽짾짿쨀쨁쨂쨃쨄쨅쨆쨇쨈**쨉쨊쨋쨌 쨍쨎쨏쨐쨑쨒쨓쨔쨕쨖쨗쨘쨙쨚쨛쨜쨝쨞쨟쨠쨡쨢쨣쨤쨥쨦짯쨨쨩쨪쨫쨬쨭쨮쨯쨰쨱쨲쨳짼쨵쨶쨷쨸쨹쨺쨻쨼쨽쨾쨿쩀 쩁쩂쩃쩄쩅쩆쩇쩈쩉쩊쩋쩌쩍쩎쩏쩐쩑쩒쩓쩔쩕쩖쩗쩘쩙쩚쩛쩜쩝쩞쩟쩠쩡쩢쩣쩤쩥쩦쩧쩨쩩쩪쩫쩬쩭쩮쩯쩰쩱펢쩳쩴 쩵쩶쩷쩸쩹쩺쩻쩼쩽쩾쩿펰쪁쪂쪃쪄쪅쪆쪇쪈쪉쪊쪋쪌쪍쪎쪏쪐쪑쪒쪓쪔쪕쪖쪗쪘쪙쪚쪛쪜쪝쪞쪟쪠쪡쪢쪣폔쪥쪦쪧쪨 쪩쪪쪫쪬쪭쪮쪯쪰폡쪲쪳쪴쪵쪶쪷쪸쪹쪺쪻쪼쪽쪾쪿쫀쫁쫂쫃쫄쫅쫆쫇쫈쫉쫊쫋쪾쫍쫎쫏쫐쫑쫒쫓쫔쫕쫖쫗쫘쫙쫚쫛쫜 쫝쫞쫟쫠쫡쫢쫣쫤쫥쫦쫧쫨쫩쫪쫫쫬쫭쫮쫯쫰쫱쫲쫳퐤쫵쫶쫷쫸쫹쫺쫻쫼쫽쫾쫿쬀쬁쬂쬃쬄쬅쬆쬇쬈좽좾쬋쬌쬍쬎쬏쬐 쬑쬒쬓쬔쬕쬖쬗쬘쬙쬚쬛쬜쬝쬞쬟쬠쬡쬢쬣쬤쬥쬦쬧쬨쬩쬪쬫쬬쬭쬮쬯쬰쬱쬲쬳쬴쬵쬶쬷쬸쬹쬺쬻쬼쬽쬾쬿쭀쭁쭂쭃쭄 쭅쭆쭇쭈쭉쭊쭋쭌쭎쭎쭏쭐쭑쭒쭓쭔쭕쭖쭗쭑쭙쭚쭛쭜쭝쭟쭟쭟쭟쭟쭟쬒쭥쭦쭧쭨쭪쭪쭫쭼쭭쭯쭯쭰쭱쭲쭳쭴쭨쭶쮯쭸 쭹쭺쭻쭼쭽쭾쭿풰쮁쮂풳풴풵쮆쮇쮈쮉쮊쮋쮌쮍풾풿퓀퓁쮒쮓쮔쮕쮖쮗쮘쮙쮚쮛쮜쮝쮞쮟쮠쮡쮢쮣쮤쮥쮦쮧쮨쮩쮪쮫쮬 쮭쮮쮯쮰쮱쮳쮳쮤쮵쮶쮷쮸쮹쮺쮻쮼쮽쮾쮿쯀쯁쯂쯃쯄쯅쯆쯇쯈쯉쯊쯋쯌쯍쯎쯏쯐쯑쯒쯓쯔쯕쯖쯗쯘쯙쯚쯛쯜쯝쯞쯟쯠 쯡쯢쯣쫌쯥쯦쭛쯨쯩쯪쯫쯬쯭쯮쭣쯰쬑쯲쬓쯴쯵쯶쯷쯸쯹쯺쯻쬜쯽쬞쯿쬠찁찂쬣찄찅찆찇찈찉찊찋찌찍찎찏찐찑찒찓찔 찕찖찗찘찙찚찛찜찝찞찟찠찡찢찣찤찥찦찧가착찪찫찬찭찮찯찰찱찲찳찴찵찶찷참찹찺찻찼창찾찿챀챁챂챃채챽챆챇챈 챉챊챋챌챍챎챏챐챑챒챓챔챕챖챗챘챙챚챛챜챝챞챟챠챡챢챣챤챥챦챧챨챩챪챫챬챭챮챯챰챱챲챳챴챵챶챷챸챹챺챻챼 챽챾챿첀첁첂첃첄첅첆첇첈첉첊첋첌첍첎첏첐첑첒첓첔첕첖첗처척첚첛천첝첞첟철첡첢첣첤첥첦첧첨첩첪첫첬청첮첯첰 첱첲첳체첵첶첷첸첹첺첻쳴첽첾첿쳀쳁쳂쳃쳄쳅쳆쳇쳈쳉쳊쳋쳌쳍쳎쳏쳐쳑쳒쳓쳔쳕쳖쳗쳘쳙쳚쳛쳜쳝쳞쳟쳠쳡쳢쳣쳤 쳥첮쳧쳨쳩쳪쳫쳬쳭쳮쳯쳰쳱쳲쳳쳴쳵쳶쳷쳀쳹쳂쳃쳼쳅쳾쳿쳈쳉촂촃촄쳍촆쳏초촉촊촋촌촍촎촏촐촑촒촓촔촕촖촗촘 촙촚촛촜총촞춪춬촡츺촟촤촥촦촧촨촩촪촫촬촭촮촯촰촱촲촳촴촵촶촷촸촹촺촻촼촽촾촿쵀쵁쵂쵃쵄쵅쵅쵇쵘쵉쵊쵋쵌 쵍쵎쵏쵐쵑쵒쵓쵔쵕쵖쵗쵘쵙쵚쵛츼쵝쵞쵟쵠쵡쵢쵣쵤쵥쵦쵧쵨쵩쵪쵫쵬쵭쵮쵯쵰쵱쵲쵳쵴쵵쵶쵷쵸쵹쵺쵻쵼쵽쵾쵿춀 춁춂춃춄춅춆춇춈춉춊춋춌춍춎춏춐춑춒춓추춬춖춗춘춚춚춛출춝춞춟춠춡춢춣춖춥춦춧춠충춫춫춬춭춡춫춰춱춲춳춴

춵춶춷췈춹춺춺춾춾춾췀췸췭췻췄췅췆췇춱췉춵췇췌췐췎췠췐췑췑췔췕췖췗췘췙췚췛췐췐췠췟췠췟췢췍췍췔췙쳊칚 췩췪췫췬췭췮췯췰췱췲췳췴췵췶췷췸췹췺췻췼췽췾췿츀츁츂츃츄츅츆츇츈츊츊츋츌츍츎츏츐츑츒츓츔츕츖츗즀츙츛츛츜 츝츞츟츠측츢츣츤츥츦츧츨츩츪츫츬츭츮츯츰츱츲츳츴층츶츷츸츹츺츻츼츽츾츿칀칁칂칃칄칅칆칇칈칉칊칋칌칍칎칏칐 칑칒칓칔칕칖칗치칙칚칛친칝칞칟칠칡칢칣칤칥칦칧침칩칪칫칬칭칮칯칰칱칲칳캬칵칶칷칸칹칺칻칼칽칾칿캀캁캂캃캄 캅캆캇캈캉캊캋캌캍캎캏**개캑캒캓캔캕캖**캗캘캙캚캛캜캝캞캟캠캡캢캣캤캥캦캧캨캩캪캫캬캭캮캯캰캱캲캳캴캵캶캷캸 캹캺캻캼캽캾캿컀컁컂컃컄컅컆컇컈컉컊컋컌컍컎컏컐컑컒컓컔컕컖컗컘컙컚컛컜컝컞컟컠컡컢컣커컥컦컧컨컩컪컫컬 컭컮컯컰컱컲컳컴켭컶컷컸컹컺컻컼컽컾컿게켁켂켃퀜켅켆켇켈켉켊켋켌켍켎켏퀨켑켒켓켔켕켖켗켘켙켚켛켜켝켞켟켠 켡켢켣켤켥켦켧켨켩켪켫켬켭켮켯컜켱켲켳켴켵켶켷게켹켺켻켼켽켾켿콀콁콂콃콄콅콆콇콈콉콊콋콌콍콎콏콐콑콒콓궆 콕콖콗콘콙콚콛콜콝콟콠콡콢콣콤콥콦콧콨콩콪콫콬콭콮콯콰콱콲콳콴콵콶콷뢀콹콺콻콼콽콾콿쾀쾁쾂쾃쾄쾅쾆쾇쾈 쾉쾊쾋쾌쾍쾎쾏쾐쾑쾒쾓쾤쾕쾖쾗쾘쾙쾚쾛쾜쾝쾞쾟쾠쾡쾢쾣쾤쾥쾦쾣킈쿀쾪쾫쾬쾭쾮쾯쿀쾱쾲쾳쾴쾵쾶쾷쾸쾹쾺쾻쾼 쾽쾾쾿쿀쿁쿂쿃쿄쿅쿆쿇쿈쿉쿊쿋쿌쿍쿎쿏굜굝쿒쿓쿔쿕쿖쿗쿘쿙쿚쿛쿜굩쿞쿟쿠쿡쿢쿣쿤쿥쿦쿧쿨쿩쿪쿫쿬쿭쿮쿯쿰 쿱쿲쿳쿴쿵쿶쿷쿸쿹쿺쿻쿼쿽쿾쿿퀀퀁퀂퀃퀄퀅퀆퀇퀈퀉퀊퀋퀌퀍퀎퀏퀐퀑퀒퀓퀔퀕퀖퀗퀘퀙퀚퀛퀜퀝퀞퀟퀠퀡퀢퀣퀤 퀥퀦퀧퀨퀩퀪퀫퀬퀭퀮퀮퀰퀱퀲퀳퀴퀵퀶퀷퀸퀹퀺퀻퀼퀽퀾퀿큀큁큂큃큄큅큆큇큈큉큊큋큄큍큎큏큐큑큒큓큔큕큖큗큘 큙큚큛큜큝큞큟큠큡큢큣큤큥큧큧큨큩큪큫그큭큮큯큰큱큲큳클큵큶큷큸큹큺큻큼큽큾큿킀킁킂킃킄킅킆킇킈믝킊킋킌 킍킎킏킐킑킒킓킔킕킖킗킘킙킚킛킜킝킞킟킠킡킢킣키킥킦킧킨킩킪킫**킬킭킮킯킰킱킲킳킴킵킶**킷킸킹킺킻킼킽킾킿타 <u>탁탂탃탄탆탆탈탉탊퇇탌탍탎탏탐탑탒탓탔탕탖탗탘퇕퇖탛턔택탞탟턘탡탢퇟턜탥퇢탧탨탩탪탫턤턥탮탯탰탱탲탳탴</u> 탵탶탷탸탹탺탻탼탽탾탿턐턁턂턃턄턅턆턇턈턉턊턋턌턍턎턏턐턑턒턓턔턕턖턗턘턙턚턛턜턝턞턟턠턡턢턣턤턥턦턧턨 턩턪턫턬턭턮턯터턱턲턳턴턵턶텯털턹턺턻턼턽턾턿텀텁텂텃텄텅텆텇텈텉텊텋테톅텎텏텐텑텒텓톌텕텖텗텘텙텚텛템 텝텞텟텠텡텢텣텤텥텦텧텨텩텪텫텬텭텮텯텰텱텲텳텴텵텶텷텸텹텺텻텼텽텾텿텰톁톂톃톄톅톆톇뎬톉톊톋톌톍톎톏톐 톑톒톓톔톕톖톗톘톙톚톛톜톝톞톟토톡톢톣**톤톥톦톧톨톩톪톫톬톭톮톯톰**톱톲톳톴통톶톷톸톹톺톻톼톽톾톿퇀퇁퇂퇃퇄 퇅퇆퇇퇈퇉퇊퇋퇌퇍퇎퇏퇐퇑퇒퇓퇔퇕퇖퇗퇘퇙퇚퇛퇜퇝퇞퇟퇠퇡퇢퇣퇤퇥뢢퇧퇨퇩퇪퇫퇬퇭퇮퇯퇰됕퇲퇳퇴퇵퇶퇷퇸 **퇹퇺퇻퇼퇽퇾퇿툀툁**툂툃툄툅툆툇툈툉툊툋툌툍툎툏툐툑툒툓툔툕툖툗툘툙툚툛툜툝툞툟툠툡툢툣툤툥툦툧툨툩툪둏투 툭툮툯툰툱툲툳툴툵툶툷툸툹툺툻툼툽툾툿퉀퉁퉃퉃퉄툳퉆퉇퉈퉉퉊퉋퉌퉍퉎퉏퉐퉑퉒퉓퉔퉕퉖퉗퉘퉙퉚퉛퉜퉝퉞퉟퉠 퉡퉢퉣퉤퉥퉦뒋퉨뒍뒎뒏뒐퉭퉮퉯뒔퉱퉲퉳**퉴**퉵퉶퉷퉸뒝퉺퉻퉼퉽퉾뒣튀튁뒦튃튄튅튆튇튘튉튋튌튍튎튏튐튑튒튓튔 튕뒺뒼뒼뒾튗튜튝튞튟튠튢튢튣튤튥튦튧튨튩튪튫튬튭튮튯튰튱튳튷튴튵튶튷트틐튺튻튼튽튾튿틐틁틂틃틄틅틆틇톰 틉틊틋틌틍틎틏틐틑틒틓틔틕틖틗틘틙틚틛틜틝틞틟틠틡틢틣틤튑틦틧딌틩퉞틫틬틭틮틯티틱틲틳틴틵틶틷틸틹틺틻틼 틽틾틿팀팁팂팃팄팅팆팋팈팉팊팋파팍팎팏판팑팒팓팔팕팖팗팘팙팚팛팜팝팞팟팠팡팢팣팤팥팦팧퍠퍡팪팫퍤퍥팮팯퍨 팱팲팳팴팵팶팷팸팹팺팻팼팽팾팿퍀퍁퍂퍃퍄퍅퍆퍇퍈퍉퍊퍋퍌퍍퍎퍏퍐퍑퍒퍓퍔퍕퍖퍗퍘퍙퍚퍛퍌퍝퍞퍟퍠퍡퍢퍣퍤 퍥퍦퍧퍨퍩퍪퍫퍬퍭퍮퍯퍰퍱퍲퍳퍴퍵퍶퍷퍸퍹퍺퍻퍼퍽퍾퍿펀펁펂펃펄펅펆펇펈펉펊펋펌펍펎펏펐펑펒펓펔펕펖펗페 펙펚펛펜펝펱펠펡펢펣펤펥펦펧펨펩펪펫펬펭펮펯펰펱펲펳펴펵펶펷편펹펺펻펼펽펾펿폀폁폂폃폄폅폆폇폈평폊폋폌 폍폎폏폐폑폒폓폔폕폖폗폘폙폚폛폜폝폞폟폠폡폢폣폤폥폦폧폨폩폪폫**포**폭폮폯폰폱폲폳폴폵폶폷폸폹폺폻폼폽폾폿퐀 퐁퐂퐃퐄퐅퐆퐇퐈퐉퐊퐋퐌퐍퐎퐏퐐퐑퐒퐓퐔퐕퐖퐗퐘퐙퐚퐛퐜퐝퐞퐟퐠퐡퐢퐣퐤퐥퐦퐧퐨퐩퐪퐫퐬퐭퐮퐯퐰퐱퐲퐳퐴 퐵퐶퐷퐸퐹퐺퐻퐼퐽퐾퐿푀푁푂푃푄푅푆푙푈푉푊푋푌푍푎푏푐푑푒푓푔푕푖푗푘푙푚푛뚀푝푞푟푠푡푡푣푤푥푦푧푨 푩푪푫푬푭푮푯푰푱푲푳푴푵푶푷푸푹푺푻푼푽푾푿풀풁풂풃풄풅풆풇품풉풊풋풌픙풏풏풐풑풒풏풔풕풖풗풘풙풚풛풜 풝풞풟풠풡풢풣풤풥풦풧풨풩풪풫풬풭풮풯풰풱풲풳풴풵풶풷풸풹풺풻풼풽풾풿퓀퓁퓂퓃퓄퓅퓆퓇퓈퓉퓊**퓋퓌퓍퓎퓏퓐** 퓑퓒퓓퓔퓕퓚퓗퓚퓙퓚퓛퓜퓝퓞퓟퓠퓡퓢퓧퓔퓥퓦퓧퓨퓩퓪퓫퓬퓮퓮퓯퓰퓱퓲퓳퓶퓵퓶퓷퓸퓸퓺퓻퓼퓽퓿퓿픀픁픂퓿프 픅픆픇굔픉픊픋플픍픎픏픐픑픒픓픔픕픖픗픘픙픚픛픜픝픞픟피픡픢픣픤픥픦픧픨픩픪픫픬픭픮픯픰픱픲픳픴픵픶픷픸 픹픺픻피픽픾픿핀핁핂핃필핅핆핇핈핉핊핋핌핍핎<u>핏</u>핐핑핒핓핔핕핖핗하학핚핛한핝핞핟할핡핢핣핤핥핦핧핡합핪햣핬 항핮핯핰핱핲핯해핵핶핷핸핹핺핻햌핽핾핿햀햁햂햃햄햅햆햿했헁햊햋햌햍햎햏햐햑햒햓햔햕햖햗햘햙햚햛햜햝햞햟햠

햡햢햣햤향햦햧햨햩햪햫햬햭햮햯햰햱햲햳햴햵햶햷햸햹햺햻햼햽햾햿헀헁헂헃헄헅헆헇허헉헊헋헌헍헎헏헐헑헒헓헔 헕헖헗험헙헚헛헜헝헞헟헠헡헢헣헤헥헦헧헨혡헽헫혥헮헯헰헱헲헳헴헵헶헷헸헹헺헻헼헽헾헿쳑텩혂혃쳔혅혆혇혈 혉혊혋혌혍혎혏혐뗩혒혓쪘형혖혗혘혙혚혛혜혝혞혟혠혡혢혣혤혥혦혧혨롍혪혫혬혭혮혯혰혱혲혳혴혵혶혷호혿혺혻혼 혽혾혿홀훍홂훏홄홅홆훓흠홉홊홋홌홍훚홏홐홑홒홓촤봑홖홗봔홙홚홛활홝홞홟홠홡홢홣홤홥홦홧홨뢍홪홫홬홭홮홯홰 홱좪왟홴홵홶홷홸홹꽮홻홼홽홾홿횀횁횂횃횄좽횆횇횈횉횊횋쾨횎횎횐흱횒횓횔횕횖훫흸횙횚횛횜횝횞횟힀죙횢횣횤 횥횦횧됴횩횪횫횬횭횮횯횰횱횲횳횴횵횶횷횸횹횺횻횼횽횾횿훀훁훂훃우웈춖욳쭌툱훊훋훌훍훎욻훐훑훒훓욹훕웂읏웄 훙훛붗훜훝훞쿻훠훡쭦훣훤훥훦훧퉐훩훪훫훬훭훮훯훰훱훲훳훴훵훶훷훸훹훺훻넦훽쮂웫휀휁휂휃퉬휅휆뷃휈휉휊휋휌 뤱휎휏휐휑휒휓뤸휕휖뤻쮜휙퀶휛휜륁휞휟퓔휡휢퀿휤휥큂휧휨헙휪휫휬휭퓢휯휰휱휲휳츄휵휶휷휸휹휺휻휼휽휾휿흀

```
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 = [['¬', '¬'], ['ㄴ'], ['ㄴ'], ['ㄹ'], ['ㅁ'], ['ㅂ', 'ㅃ'], ['ㅅ', 'ㅆ'], ['ㅇ'], ['ㅈ', '짜'], ['ㅊ'], ['¬'],
              ['⊑'], ['≖'], ['ㅎ']]
Consonants = [[''], ['¬', 'ㄲ', 'ಒ'], ['ㄴ', 'ㄶ'], ['ㄷ'], ['ㄹ', 'ಏ', '訑', 'ඪ', 'མ', 'ஊ', 'ஊ', '혀'], ['ㅁ'],
               ['ㅂ', 'ㅄ'], ['ㅅ', 'ㅆ'], ['ㅇ'], ['ㅈ', 'ㅉ'], ['ㅊ'], ['ㅋ'], ['ㅌ'], ['ㅍ'], ['ㅎ']]
Vowel = ['ㅏ', 'ㅐ', 'ㅑ', 'ㅒ', 'ㅓ', 'ㅖ', 'ㅕ', 'ㅖ', 'ㅗ', 'ㅘ', 'ㅚ', 'ㅛ', 'ㅜ', '柌', 'ᆌ', 'ㅟ', 'ㅡ', 'ㅢ', 'ㅣ']
Z = ['¬', '∟', '⊏', 'ㄹ', 'ㅁ', 'ㅂ', 'ㅆ', 'ㅇ', 'ㅈ', 'ㅊ', '¬', '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]
```

Z:

```
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
    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)):
        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 더
한다.
       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 I 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!')
```