My Python script *Hangul.py* basically generates a LaTeX file with tikz code for generating several types of Hangul character related images. These images range from components of a Hangul character corresponding to vowel and consonant markers, to strings of characters aligned into rows of certain length. The tikz code for each image type is written by a different command within the Python script. The first command *vowel* writes tikz code for an upright, black bar with randomly generated smaller bars sprouting from the sides of the larger bar. The second command *consonant* randomly selects from a choice of fourteen different code lines, each generating a different graph that represents a consonant in Hangul. The third command *character* combines a vowel graph with a consonant graph by defining two sub-environments within the tikz environment. The first sub-environment randomly either rotates the vowel graph at a 90 degree angle or maintains its upright position. The second sub-environment stretches the consonant graph along either the x or y axis, depending on whether the vowel graph is upright or rotated, following general rules of combining these graphs in Hangul.

```
def character():
'''combine vowels and consonants'''
upright = '\t\\begin{scope}[shift={(4,0)}]\n'
acostado = '\t\\begin{scope}[shift={(18,-4)}, rotate=90]\n'
position = random.choice([upright,acostado])
stretchX = '\t\\begin{scope}[shift={(-6.4,1)}]\n'+'\t\\pgftransformxscale{1.7}\n'
stretchY = '\t\\begin{scope}[shift={(-1,-6.5)}]\n'+'\t\\pgftransformyscale{1.7}\n'
f.write(position)
vowel()
f.write('\t\\end{scope}\n')
if position == upright:
f.write(stretchY)
else:
f.write(stretchX)
consonant()
f.write('\t\\end{scope}\n')

character()
f.write('\n')
```

The *word* and *characters* commands do the same thing in different ways. They both write code that generates a sequence of Hangul characters with a numerical input for the number of characters desired. *word* does this by inserting the output of the *character* command in another sub-environment and performing this action a set number of times, with each iteration featuring a horizontal shift in coordinates. *characters* is a rewrite of *character* with a set number of iterated characters, each with a horizontally shifted coordinates. The *text* command is a rewrite of the *word* command that writes tikz code for several strings of Hangul characters separated by spaces. It takes two inputs, a maximum number for string length and number of words desired. The code basically takes uses a *for* loop to generate character code a random number of times, before inserting a space and starting the process over again. The insertion of a space is also performed within a *for* loop for a set number of times. The *page* command does the same thing as the *text* command, except that several *if* lines are inserted to induce a maximum horizontal length of character strings with spaces. Once the maximum length is reached, the string continues on another line down, starting at the original x coordinate. The maximum length is set as the third argument of the command. The resulting image appears to be a page of Hangul text written from left to right, top to bottom, with spaces.

```
def page(nmChar,nmWord,nmLine):
yOffset = 0
xOffset = 0
for x in range(nmWord):
for x in range(random.randint(1,nmChar)):
f.write('\\begin{scope}[shift={('+str(xOffset)+','+str(yOffset)+')}]\n')
character()
f.write('\\end{scope}\n')
f.write('\n')
xOffset += 12
if xOffset >= (12*nmLine):
xOffset = 0
yOffset -= 14
f.write('\\begin{scope}[shift={('+str(xOffset)+','+str(yOffset)+')}]\n')
f.write('\\end{scope}\n')
f.write('\n')
xOffset += 12
if xOffset >= (12*nmLine):
xOffset = 0
yOffset -= 14

page(9,random.randint(1,9),7)
```

거므표요크두혀

으 표호저여우

치티효 챠됴

츄됴죠크다느