In [ ]:
```
import nltk
from nltk.stem import PorterStemmer, LancasterStemmer, SnowballStemmer, RegexpSt
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.tokenize import word_tokenize


# abstract paragraph
paragraph = "This literature study delves into the essence of design within soft

# Tokenize the paragraph
# tokens = word_tokenize(paragraph)
tokens = ['is','soft','part','perform','struct','tonic','bubblegum']

# Apply Snowball Stemmer
snowball_stemmer = SnowballStemmer('english')
snowball_stemmed_tokens = [snowball_stemmer.stem(token) for token in tokens]
print("Snowball Stemmer: ", snowball_stemmed_tokens)

# Apply Porter Stemmer
porter_stemmer = PorterStemmer()
porter_stemmed_tokens = [porter_stemmer.stem(token) for token in tokens]
print("Porter Stemmer: ", porter_stemmed_tokens)

# Apply Lancaster Stemmer
lancaster_stemmer = LancasterStemmer()
lancaster_stemmed_tokens = [lancaster_stemmer.stem(token) for token in tokens]
print("Lancaster Stemmer: ", lancaster_stemmed_tokens)

# Apply Lemmatization
lemmatizer = WordNetLemmatizer()
lemmatized_tokens = [lemmatizer.lemmatize(token) for token in tokens]
print("Lemmatization: ", lemmatized_tokens)


# Apply Regexp Stemmer
regexp_stemmer = RegexpStemmer('ing$|s$|ed$|ly$|ment$|ness$|er$|est$|able$|ic$|a
regexp_stemmed_tokens = [regexp_stemmer.stem(token) for token in tokens]
print("Regexp Stemmer: ", regexp_stemmed_tokens)
```

```
Snowball Stemmer:  ['is', 'soft', 'part', 'perform', 'struct', 'tonic', 'bubblegu
m']
Porter Stemmer:  ['is', 'soft', 'part', 'perform', 'struct', 'tonic', 'bubblegu
m']
Lancaster Stemmer:  ['is', 'soft', 'part', 'perform', 'struct', 'ton', 'bubbleg']
Lemmatization:  ['is', 'soft', 'part', 'perform', 'struct', 'tonic', 'bubblegum']
Regexp Stemmer:  ['i', 'soft', 'part', 'perform', 'struct', 'ton', 'bubblegum']
```