# Ex1a: Text Generation using N-gram Language model

**Learning Objective:**

Implement an N-Gram–basedpredictivetext modelusingbigram andtrigram probability distributions. This helps to understand how contextual probability influences next-word prediction.

Steps:

1. Initialize the environment by installing and importing the Natural Language Toolkit (nltk). Download the **Brown Corpus** (for text data) and the **Universal Tagset** (for simplified Part-of-Speech tags).
2. Extract raw sentences from the Brown Corpus using brown.sents().
3. **Tokenization:** Convert all words to lowercase and flatten the nested list into a single list of tokens to verify the total volume of training data.
4. Build an N-Gram Model for Bigram and Trigram.
5. Predict Next Word and display Top-5 Suggestions
6. Test Bigram & Trigram Prediction on certain words.

**Program:**

```
import nltk

from nltk.corpus import brown

from collections import defaultdict, Counter

import random

from collections import defaultdict, Counter

import nltk

from nltk.corpus import brown

nltk.download('brown')

words = [word.lower() for word in brown.words() if word.isalpha()]

def build_ngram_model(words, n=3):

  ngram_counts = defaultdict(Counter)

  for i in range(len(words) - n + 1):

    context = tuple(words[i:i+n-1])

    target = words[i+n-1]

    ngram_counts[context][target] += 1

  return ngram_counts


def show_next_word_counts(context, model):

 context = tuple(context.lower().split())

 if context not in model:

 print("No prediction available")

 return
 total_count = sum(model[context].values())
 print(f"Context: {' '.join(context)}")
 print(f"TOTAL count: {total_count}\n")
 print("Next word counts:\n")
 for word, count in model[context].most_common():
 print(f"{word:<15} : {count}")
# Example usage
N = 2
ngram_model = build_ngram_model(words, N)
context = "Apple"
show_next_word_counts(context, ngram_model)
```

**Output:**

```
Context: apple
TOTAL count: 9
Next word
counts:
pie : 2
tree : 2
trees : 1
pies : 1
cider : 1
the : 1
and : 1
```

**Learning Outcome:**

Uponcompletionof this experiment, the bigram and trigram probability distributions for next-word prediction was implemented and evaluated using the Brown Corpus.