EXPT NO: 1 B)	HIDDEN MARKOV MODEL (HMM) BASED PREDICTIVE TEXT SYSTEM
DATE: 18/02/26	

AIM

To implement a predictive text system using Hidden Markov Model (HMM), enabling students to understand contextual probability, POS transitions, and next-word prediction using the Brown Corpus.

ALGORITHM/ STEPS

- **STEP 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).
- **STEP 2:** Extract raw sentences from the Brown Corpus using brown.sents()
- **STEP 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.
- **STEP 4:** Load the tagged version of the corpus (brown.tagged_sents) which provides the "Hidden States" (POS tags) linked to the "Observations" (words)
- **STEP 5:** Train the HMM Model. Use the HiddenMarkovModelTrainer to perform Supervised Learning. Calculate the internal parameters of the HMM: Initial State Probabilities, Transition Probabilities, Emission Probabilities
- **STEP 6:** Build POS Transition Probabilities
- **STEP 7:** Build Emission Probabilities

- **STEP 8:** Define Prediction Function
- **STEP 9:** Test the Model\
- **STEP 10:** Verify POS prediction on certain words.

CODE IMPLEMENTATION

```
import nltk
from nltk.corpus import brown
from nltk.tag.hmm import HiddenMarkovModelTrainer
import string

# Download necessary NLTK data: 'brown' corpus and 'universal_tagset'.
# These downloads are required for the HMM POS tagging task.
nltk.download('brown')
nltk.download('universal_tagset')

# --- Data Preparation ---

# Load sentences from the Brown corpus.
sentences = brown.sents()

# Tokenize the corpus, convert words to lowercase, and filter out
punctuation.
tokens = []
for sent in sentences:
    for word in sent:
        word = word.lower()
        if word not in string.punctuation:
            tokens.append(word)

# Print introductory information and total token count.
print("=======================================")
print(" HMM POS Tagging using Brown Corpus ")
print("=======================================")
print(f"Total Tokens in Corpus : {len(tokens)}")

# Load tagged sentences from the Brown corpus using the 'universal' tagset.
# Convert words to lowercase and remove punctuation from tagged sentences.
tagged_sentences = brown.tagged_sents(tagset='universal')
tagged_sentences = [
    [(word.lower(), tag) for word, tag in sent if word not in
string.punctuation]
    for sent in tagged_sentences
```

```
]
print(f"Total Tagged Sentences : {len(tagged_sentences)}")

# --- HMM Model Training ---

# Initialize the Hidden Markov Model (HMM) trainer.
trainer = HiddenMarkovModelTrainer()

# Train the HMM model using the supervised (pre-tagged) sentences.
# This step learns transition and emission probabilities from the data.
hmm_model = trainer.train_supervised(tagged_sentences)
print("\nHMM Model Trained Successfully")

# --- Displaying Model Parameters ---

# Access and display sample transition probabilities.
# Transition probability: P(tag_i | tag_{i-1})
transition_prob = hmm_model._transitions
print("\nSample POS Transition Probabilities:")
for state in list(transition_prob.keys())[:5]: # Displaying first 5 states
for brevity
    print(f"{state} -> {transition_prob[state]}")

# Access and display sample emission probabilities.
# Emission probability: P(word_i | tag_i)
emission_prob = hmm_model._outputs
print("\nSample Emission Probabilities:")
for state in list(emission_prob.keys())[:5]: # Displaying first 5 states for
brevity
    # _freqdist.most_common(5) shows the 5 most probable words for that tag
    print(f"{state} -> {emission_prob[state]._freqdist.most_common(5)}")

# --- POS Prediction Function ---

# Define a function to predict POS tags for a given sentence.
def predict_pos(sentence):
    # Convert all words in the input sentence to lowercase for consistency.
    sentence = [word.lower() for word in sentence]
    # Use the trained HMM model to tag the sentence.
    return hmm_model.tag(sentence)

# --- User Interaction for Sentence Tagging ---

# Prompt the user to enter a sentence for POS tagging.
user_sentence = input("\nEnter a sentence for POS tagging:\n")
# Split the input sentence into individual words.
user_words = user_sentence.split()
# Predict POS tags for the user's sentence.
```

```
predicted_tags = predict_pos(user_words)

# Print the predicted POS tags in a formatted way.
print("\n----------------------------------------")
print(" POS Prediction for Given Sentence ")
print("----------------------------------------")
for word, tag in predicted_tags:
    print(f"{word:<12} -> {tag}")

# --- User Interaction for Individual Word Verification ---

# Prompt the user to enter words to verify their POS tags individually.
user_test_words = input(
    "\nEnter words to verify POS (space-separated):\n"
).split()

# Print the POS tags for each individual word entered by the user.
print("\n----------------------------------------")
print(" POS Verification on Individual Words ")
print("----------------------------------------")
for word in user_test_words:
    # Tag each word individually using the HMM model.
    # [0][1] extracts the tag from the single-word tagging result.
    tag = hmm_model.tag([word.lower()])
    print(f"{word:<12} -> {tag[0][1]}")
```

OUTPUT

```
[nltk_data] Downloading package brown to /root/nltk_data...
[nltk_data]   Unzipping corpora/brown.zip.
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Unzipping taggers/universal_tagset.zip.
========================================
 HMM POS Tagging using Brown Corpus
========================================
Total Tokens in Corpus : 1034378
Total Tagged Sentences : 57340

HMM Model Trained Successfully

Sample POS Transition Probabilities:
DET -> <MLEProbDist based on 136699 samples>
NOUN -> <MLEProbDist based on 241376 samples>
ADJ -> <MLEProbDist based on 80599 samples>
VERB -> <MLEProbDist based on 176495 samples>
ADP -> <MLEProbDist based on 144471 samples>

Sample Emission Probabilities:
DET -> [('the', 69968), ('a', 23070), ('his', 6957), ('this', 5144), ('an', 3726)]
NOUN -> [('time', 1597), ('man', 1203), ('af', 995), ('years', 949), ('way', 899)]
ADJ -> [('other', 1702), ('new', 1635), ('first', 1031), ('many', 996), ('more', 989)]
VERB -> [('is', 10108), ('was', 9815), ('be', 6374), ('had', 5132), ('are', 4394)]
ADP -> [('of', 36410), ('in', 20866), ('to', 11158), ('for', 9481), ('with', 7286)]
```

```
Enter a sentence for POS tagging:
the world
/usr/local/lib/python3.12/dist-packages/nltk/tag/hmm.py:335: RuntimeWarning: overflow encountered in cast
  O[i, k] = self._output_logprob(si, self._symbols[k])

----------------------------------------
 POS Prediction for Given Sentence
----------------------------------------
the          -> DET
world        -> NOUN

Enter words to verify POS (space-separated):
beautiful

----------------------------------------
 POS Verification on Individual Words
----------------------------------------
beautiful    -> ADJ
```

RESULT

Upon completion of this experiment, HMM based model was built to predict the next word with POS-based suggestions, and evaluated the effectiveness of context-driven predictive text generation using the Brown Corpus.