```
class CYKParser:
    def __init__(self):
        """
        Initialize CYK Parser with comprehensive grammar rules
        """
        self.grammar = {
            'S': [['NP', 'VP'], ['VP']],
            'NP': [['Det', 'N'], ['Proper'], ['N'], ['Det', 'Adj', 'N']],
            'VP': [['V'], ['V', 'NP'], ['V', 'Adv'], ['V', 'NP', 'Adv']],
            'Det': [['the'], ['a'], ['an'], ['this'], ['that']],
            'N': [['dog'], ['cat'], ['mouse'], ['man'], ['ball'], ['book'], ['tree'], ['house']],
            'Proper': [['John'], ['Mary'], ['Tom'], ['Sarah'], ['Emma'], ['David'], ['Anna']],
            'V': [['chased'], ['saw'], ['liked'], ['ran'], ['jumped'], ['read'], ['wrote'], ['played']],
            'Adv': [['quickly'], ['slowly'], ['carefully'], ['happily'], ['quietly']],
            'Adj': [['big'], ['small'], ['red'], ['blue'], ['happy'], ['sad']]
        }

    def parse(self, sentence):
        """
        CYK Parsing Algorithm Implementation
        """
        if isinstance(sentence, str):
            sentence = sentence.split()
        n = len(sentence)
        table = [[set() for _ in range(n-j)] for j in range(n)]
        derivation_table = [[[] for _ in range(n-j)] for j in range(n)]
        for i, word in enumerate(sentence):
            for nt, productions in self.grammar.items():
                for prod in productions:
                    if len(prod) == 1 and prod[0] == word:
                        table[0][i].add(nt)
                        derivation_table[0][i].append((nt, [word]))
        for j in range(1, n):
            for i in range(n-j):
                for k in range(j):
                    for nt, productions in self.grammar.items():
                        for prod in productions:
                            if len(prod) == 2:
                                left, right = prod
                                if (left in table[j-k-1][i] and right in table[k][i+j-k]):
                                    table[j][i].add(nt)
                                    derivation_path = [(left, table[j-k-1][i]), (right, table[k][i+j-k])]
                                    derivation_table[j][i].append((nt, derivation_path))
        self._print_parse_table(table, sentence)
        is_valid = 'S' in table[n-1][0]
        return is_valid, table, derivation_table

    def _print_parse_table(self, table, sentence):
        """
        Visualize the CYK parsing table
        """
        print("\n--- CYK Parsing Table ---")
        print("Input Sentence:", " ".join(sentence))
        for i, row in enumerate(table):
            print(f"\nDiagonal {i}:")
            for j, cell in enumerate(row):
                print(f"  Cell [{i},{j}]: {cell}")

    def parse_manual_sentence(self, sentence):
        """
        Parse a manually provided sentence and display results
        """
```

```
        if not sentence.strip():
            print("No sentence provided.")
            return
        print(f"Sentence: '{sentence}'")
        print(f"\n=== Parsing Sentence ===")
        is_valid, table, _ = self.parse(sentence)
        print(f"\nSentence: '{sentence}'")
        print(f"Valid Parse: {is_valid}")
```

```
def main(sentence):
    """
    Parse a single sentence using the CYK Parser
    """
    parser = CYKParser()
    parser.parse_manual_sentence(sentence)
```

```
def test_sen(z=None):
    """
    Test a sentence from test_cases, auto-incrementing the index each call
    """
    # Initialize counter as function attribute if not set
    if not hasattr(test_sen, 'counter'):
        test_sen.counter = 0

    # Use provided z if given, otherwise use counter
    index = z if z is not None else test_sen.counter

    if index < len(test_cases):
        print(f"Test Case {index}")
        main(test_cases[index])
        # Auto-increment counter for next call
        test_sen.counter = index + 1
    else:
        print("No more test cases to process.")
```

```
test_cases = [
    "the dog chased the cat",
    "John saw a mouse",
    "Mary liked the dog",
    "Tom ran quickly",
    "a cat chased",
    "the mouse John",
    "the big dog ran quickly",
    "John wrote a book carefully"
]
```

```
test_sen()
```

```
Test Case 0
Sentence: 'the dog chased the cat'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: the dog chased the cat

Diagonal 0:
  Cell [0,0]: {'Det'}
  Cell [0,1]: {'N'}
  Cell [0,2]: {'V'}
  Cell [0,3]: {'Det'}
  Cell [0,4]: {'N'}
```

```
Diagonal 1:
  Cell [1,0]: {'NP'}
  Cell [1,1]: set()
  Cell [1,2]: set()
  Cell [1,3]: {'NP'}

Diagonal 2:
  Cell [2,0]: set()
  Cell [2,1]: set()
  Cell [2,2]: {'VP'}

Diagonal 3:
  Cell [3,0]: set()
  Cell [3,1]: set()

Diagonal 4:
  Cell [4,0]: {'S'}

Sentence: 'the dog chased the cat'
Valid Parse: True
```

```
test_sen()
```

```
Test Case 1
Sentence: 'John saw a mouse'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: John saw a mouse

Diagonal 0:
  Cell [0,0]: {'Proper'}
  Cell [0,1]: {'V'}
  Cell [0,2]: {'Det'}
  Cell [0,3]: {'N'}

Diagonal 1:
  Cell [1,0]: set()
  Cell [1,1]: set()
  Cell [1,2]: {'NP'}

Diagonal 2:
  Cell [2,0]: set()
  Cell [2,1]: {'VP'}

Diagonal 3:
  Cell [3,0]: set()

Sentence: 'John saw a mouse'
Valid Parse: False
```

```
test_sen()
```

```
Test Case 2
Sentence: 'Mary liked the dog'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: Mary liked the dog

Diagonal 0:
  Cell [0,0]: {'Proper'}
  Cell [0,1]: {'V'}
  Cell [0,2]: {'Det'}
```

```
    Cell [0,3]: {'N'}

  Diagonal 1:
    Cell [1,0]: set()
    Cell [1,1]: set()
    Cell [1,2]: {'NP'}

  Diagonal 2:
    Cell [2,0]: set()
    Cell [2,1]: {'VP'}

  Diagonal 3:
    Cell [3,0]: set()

  Sentence: 'Mary liked the dog'
  Valid Parse: False
```

```
test_sen()
```

```
Test Case 3
Sentence: 'Tom ran quickly'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: Tom ran quickly

Diagonal 0:
  Cell [0,0]: {'Proper'}
  Cell [0,1]: {'V'}
  Cell [0,2]: {'Adv'}

Diagonal 1:
  Cell [1,0]: set()
  Cell [1,1]: {'VP'}

Diagonal 2:
  Cell [2,0]: set()

Sentence: 'Tom ran quickly'
Valid Parse: False
```

```
test_sen()
```

```
Test Case 4
Sentence: 'a cat chased'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: a cat chased

Diagonal 0:
  Cell [0,0]: {'Det'}
  Cell [0,1]: {'N'}
  Cell [0,2]: {'V'}

Diagonal 1:
  Cell [1,0]: {'NP'}
  Cell [1,1]: set()

Diagonal 2:
  Cell [2,0]: set()

Sentence: 'a cat chased'
Valid Parse: False
```

```
test_sen()
```

```
Test Case 5
Sentence: 'the mouse John'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: the mouse John

Diagonal 0:
  Cell [0,0]: {'Det'}
  Cell [0,1]: {'N'}
  Cell [0,2]: {'Proper'}

Diagonal 1:
  Cell [1,0]: {'NP'}
  Cell [1,1]: set()

Diagonal 2:
  Cell [2,0]: set()

Sentence: 'the mouse John'
Valid Parse: False
```

```
test_sen()
```

```
Test Case 6
Sentence: 'the big dog ran quickly'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: the big dog ran quickly

Diagonal 0:
  Cell [0,0]: {'Det'}
  Cell [0,1]: {'Adj'}
  Cell [0,2]: {'N'}
  Cell [0,3]: {'V'}
  Cell [0,4]: {'Adv'}

Diagonal 1:
  Cell [1,0]: set()
  Cell [1,1]: set()
  Cell [1,2]: set()
  Cell [1,3]: {'VP'}

Diagonal 2:
  Cell [2,0]: set()
  Cell [2,1]: set()
  Cell [2,2]: set()

Diagonal 3:
  Cell [3,0]: set()
  Cell [3,1]: set()

Diagonal 4:
  Cell [4,0]: set()

Sentence: 'the big dog ran quickly'
Valid Parse: False
```

```
test_sen()
```

```
Test Case 7
Sentence: 'John wrote a book carefully'

=== Parsing Sentence ===

--- CYK Parsing Table ---
Input Sentence: John wrote a book carefully

Diagonal 0:
  Cell [0,0]: {'Proper'}
  Cell [0,1]: {'V'}
  Cell [0,2]: {'Det'}
  Cell [0,3]: {'N'}
  Cell [0,4]: {'Adv'}

Diagonal 1:
  Cell [1,0]: set()
  Cell [1,1]: set()
  Cell [1,2]: {'NP'}
  Cell [1,3]: set()

Diagonal 2:
  Cell [2,0]: set()
  Cell [2,1]: {'VP'}
  Cell [2,2]: set()

Diagonal 3:
  Cell [3,0]: set()
  Cell [3,1]: set()

Diagonal 4:
  Cell [4,0]: set()

Sentence: 'John wrote a book carefully'
Valid Parse: False
```

```
test_sen()
```

```
No more test cases to process.
```