**BIG O NOTATION**

**(Lesson Guide)**

**I. Introduction to Big O Notation**

* **Importance of Algorithm Analysis**
  + Understanding algorithm efficiency for comparisons and informed decisions.
* **Definition of Big O Notation**
  + Standardized efficiency communication.
  + Simplifies reasoning and algorithm comparison.

**II. Basic Concepts of Big O Notation**

* Focuses on growth rate for large inputs, ignores constants.
* Expressed as "O(f(n))" for various complexities.

**III. Understanding Time Complexity**

* **Measuring Algorithm Efficiency**
  + Predicting runtime as a function of input size.
* **How to Calculate Time Complexity**
  + Counting operations and simplifying for Big O.

**IV. Common Time Complexities**

* **O(1) - Constant Time**
  + Fixed runtime, independent of input size.
* **O(log n) - Logarithmic Time**
  + Efficient search and divide-and-conquer algorithms.
* **O(n) - Linear Time**
  + Direct relationship between input size and runtime.
* **O(n^2), O(n^3), ... - Polynomial Time**
  + Nested loops and polynomial growth.

**V. Conclusion**

* **Recap of Key Points**
  + Summarizing Big O notation concepts.
* **Emphasizing the Importance of Algorithm Analysis**
  + Reinforcing its role in algorithm selection and optimization.