Comparison of Time Complexity: Linear vs Binary Search

Linear Search:

* Best Case: O(1)
* Average Case: O(n)
* Worst Case: O(n)

Binary Search:

* Best Case: O(1)
* Average Case: O(log n)
* Worst Case: O(log n)

Explanation:  
Linear search goes through each element one by one. In the worst-case scenario, it may have to check all elements in the array, making it O(n).  
Binary search works only on a sorted array. It repeatedly divides the array into halves and checks the middle element, making it much faster with a time complexity of O(log n) in average and worst cases.

Suitable Algorithm for the Given Platform

In the Java program:

* The user enters a list of items.
* The list is sorted before using binary search.
* Both linear and binary search methods are used to search for a product ID.

If the number of items is small or only a single search is needed, linear search is more suitable as it does not require sorting and works fine for small datasets.

If multiple searches are expected on the same dataset, binary search is more efficient. Though it requires the list to be sorted first (which takes O(n log n)), it performs each search in O(log n) time.

Conclusion:  
Use linear search when the dataset is small or only one search is needed.  
Use binary search when the dataset is large or when many searches are required after sorting the data once.