Searching Techniques Cheat Sheet

1. SENTINEL SEARCH

A variation of linear search where the target is placed at the end of the array as a sentinel. This avoids the need to check the boundary conditions during the search loop.

Advantages:

- Faster than basic linear search as it removes one comparison inside the loop.

Java Example:

```
int last = arr[n - 1];
arr[n - 1] = key; // Sentinel
int i = 0;
while (arr[i] != key) i++;
arr[n - 1] = last;
if (i < n - 1 || arr[n - 1] == key) return i;
```

2. PROBABILITY SEARCH (SELF-ORGANIZING SEARCH)

This technique improves linear search by rearranging elements based on access frequency.

Variants:

- Move-to-Front: Move accessed item to front.
- Transpose: Swap accessed item with its previous item.
- Count Heuristic: Reorder by frequency.

Java Example (Move-to-Front):

```
for (int i = 0; i < arr.length; i++) {
    if (arr[i] == key) {
        int temp = arr[i];
        for (int j = i; j > 0; j--) arr[j] = arr[j - 1];
        arr[0] = temp;
        return i;
    }
}
```

3. ORDERED LIST SEARCH

Optimized linear search for sorted lists. Stops early if a greater element is found.

Java Example:

```
for (int i = 0; i < arr.length; i++) {
    if (arr[i] == key) return i;
    else if (arr[i] > key) break;
}
```

SUMMARY COMPARISON

Technique	Best Use Case	Time Complexity
Sentinel Search	Slight optimization over linear search	O(n)
Probability Search	Frequently accessed elements	O(n), improves
Ordered List Search	When array is sorted	O(n), but faster