Keyword Searching, Pattern Matching & Recursion - Cheat Sheet (Theory Only)

Keyword Searching

1. Text Line Adjustment

- **Adjusts text to fit within a given width by adding spaces or breaking lines properly.**
- **Used in word processors and text formatting tools.**
- **Example Algorithm:**
- `Start`
- `Input text and maximum line width`
- `While there are words left in the text`
- `   Add words to the line until width is reached`
- `   Print the formatted line`
- `End While`
- `End`
-

Pattern Searching Algorithms

2. Linear Pattern Search

- **Checks each character in the text sequentially.**
- **Time Complexity: O(NM)** (N = text length, M = pattern length).
- **Algorithm (Brute Force Approach):**
- `Start`
- `Input Text and Pattern`
- `For i = 0 to Text Length - Pattern Length`
- `   For j = 0 to Pattern Length`
- `      If Text[i + j] ≠ Pattern[j], Break`
- `   If j = Pattern Length, Print "Pattern Found"`
- `End For`
- `End`
-

---

3. Sub-Linear Pattern Search (Efficient Methods)

- **Faster than linear search algorithms.**
- **Example: Boyer-Moore Algorithm (Skips unnecessary comparisons).**
- **Algorithm (Simplified Boyer-Moore Approach):**
- `Start`
- `Precompute shift table for pattern`
- `Align pattern with text`
- `While within text length`
- `   Compare pattern from right to left`
- `   If match, print "Pattern Found"`
- `   Else, shift pattern based on precomputed table`
- `End While`
- `End`
-

Recursion in Problem Solving

4. Towers of Hanoi

- **Moves N disks from Source to Destination using an Auxiliary Peg.**
- **Recursive Formula:** $T(N) = 2T(N-1) + 1$
- **Algorithm:**

```
Function Hanoi(N, Source, Auxiliary, Destination)
   If N = 1
      Move disk from Source to Destination
   Else
      Hanoi(N-1, Source, Destination, Auxiliary)
      Move disk from Source to Destination
      Hanoi(N-1, Auxiliary, Source, Destination)
End Function
```

---

5. Sample Generation

- **Used in random number generation, dataset sampling.**
- **Example (Recursive Random Sample Generation):**

```
Function GenerateSample(N)
   If N = 0, Return
   Generate random number
   Print it
   GenerateSample(N-1)
End Function
```

---

6. Combination Generation

- **Generates subsets from a given set.**
- **Example: Choosing K elements from an array of N elements.**
- **Algorithm:**

```
Function GenerateCombination(Array, Data, Start, End, Index, K)
   If Index = K, Print Data
   For i = Start to End
      Data[Index] = Array[i]
      GenerateCombination(Array, Data, i+1, End, Index+1, K)
```

---

7. Permutation Generation

- **Generates all possible orderings of elements.**
- **Used in password generation, anagrams, game theory.**
- **Algorithm (Backtracking Approach):**

```
Function Permute(Array, L, R)
   If L = R, Print Array
   Else
      For i = L to R
         Swap Array[L] and Array[i]
         Permute(Array, L+1, R)
         Swap Array[L] and Array[i] (Backtrack)
```

Pseudocode & Flowchart Example

Example: Towers of Hanoi Pseudocode

```
Start
Function Hanoi(N, Source, Auxiliary, Destination)
   If N = 1
      Move disk from Source to Destination
   Else
      Hanoi(N-1, Source, Destination, Auxiliary)
      Move disk from Source to Destination
      Hanoi(N-1, Auxiliary, Source, Destination)
End Function
End
```

Flowchart Symbols

Symbol	Meaning
Oval	Start/End
Rectangle	Process (Calculation)
Diamond	Decision (If/Else)
Parallelogram	Input/Output

This **Recursion & Pattern Matching Cheat Sheet** covers **keyword searching, text formatting, pattern searching (linear & sub-linear), recursion techniques (Hanoi, sample, combination, permutation), pseudocode, and flowcharts**. Let me know if you need more details!