|  |  |  |
| --- | --- | --- |
| **Problem** | **Method** | **Comments** |
|  |  |  |
|  |  |  |
|  |  |  |
| **Section 3 – Strings** | | |
| Valid Palindrome | transform(s.begin(), s.end(), s.begin(), ::tolower);  ::isalnum(int c); |  |
| Implement strStr() | Brute force, KMP, Boyer-Mooer, Rabin-Karp, Sunday? |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| **Section 7 - Search** | | |
| Search for a range | std::lower\_bound<vec.begin(), vec.end(), value)  std::upper\_bound<vec.begin(). vec.end(), value)  std::distance(vec.begin(), itr)  std::prev(itr); std::next(itr); | lower\_bound or upper\_bound uses binary search |
| Search insert position | std::lower\_bound(vec.begin(), vec.end(), value) |  |
| Search a 2D matrix | binary search |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| **Section 10 - DFS** |  |  |
| Parlindrom Paritionning | DFS (recursion) |  |
| Unique paths (in chessboard) | DFS (recursion with cache), or Math formula |  |
| Unique paths II (with obstacles) | DFS (recursion with cache), no easy math solution |  |
| N-Queeen (on N x N chessboard) | DFS with pruning | Backtrack=DFS+Pruning |
| Restore IP address | DFS with pruning |  |
| Combination Sum (select to sum to target) | DFS with pruning |  |
| Generate Parentheses | DFS (simple recursion) |  |
| Sodoku Solver | DFS |  |
| Word Search (from 2D board) |  |  |
|  |  |  |
| **Section 11 – Divide and Conquer** | | |
| Implement pow(x, n) |  |  |
| Implement sqrt(x) |  |  |
|  |  |  |
| **Section 14 – Graphs** | | |
| Clone Graph | Hashmap (std::unordered\_map) for node to node mapping. Do DFS or BFS to clone while maintaining the hash map. |  |