Here are a list of concepts, definitions, algorithms, design and analysis techniques that you should know from the class, including homeworks and midterms. This is not meant to be comprehensive. It is merely a reminder of what we need to review for the upcoming final exam.

I. Algorithms, design and analysis techniques, examples

1. Divide and Conquer
   - The three-steps of a divide and conquer algorithm
   - Examples:
     - Computing the $n$th Fibonacci number (vs. recursive)
     - MergeSort (vs. Insert sort)
     - Maximum subarrays
     - Strassen’s algorithm for matrix multiplication
     - Closest pair of points in 1D

2. Greedy Algorithms
   - Two elements of greedy algorithms
   - Examples:
     - Activity selection
     - Huffman coding
     - Minimum spanning tree: Prim’s and Kruskal’s algorithms
     - Knapsack problem (a counterexample)

3. Dynamic Programming
   - Two elements of dynamic programming
   - Examples:
     - Rod cutting
     - Matrix-chain multiplication
     - Longest common subsequence/substring
     - Knapsack problem

4. Graph algorithms
   - Elementary graph algorithms
     - Breadth-first search (BFS)
     - Depth-first search (DFS)
     - Applications
       * Topological sort of a DAG
       * Finding a sink
       * Finding connected components of a undirected graph
       * Detecting a cycle
II. NP-completeness

1. Tractable and intractable problems

2. Examples of intractable problems:
   Circuit-satisfiability (SAT), Graph-coloring, Hamiltonian-cycle (HC), Traveling-salesperson-problem (TSP), Knapsack-problem, Prime-testing, Subset-sum, Set-partition, Bin-packing, Vertex-cover.

3. Optimization problem and decision problem

4. Polynomial reduction/transformation

5. Formal definitions of P, NP, NP-complete and NP-hard

6. How to prove a problem is NP-completeness
   - The logic behind the method of proof
   - Three case studies:
     - undirected HC is NPC (from the fact that directed HC is NPC)
     - 4-color problem is NPC (from the fact that 3-color is NPC)
     - Set partition is NPC (from the fact that Subset Sum is NPC)

7. Case studies of approximate algorithms (optional):
   Bin-packing and Vertex-cover

III. Definitions, concepts and data structures

1. Growth of functions and asymptotic notations: $O(f(n)), \Omega(f(n)), \Theta(f(n))$

2. Linear recurrence relations

3. Divide-and-conquer recurrence relations

4. The master method/theorem for solving divide-and-conquer recurrence relations

5. Definitions of graph, path, connected graph, connected component, cycle, acyclic, tree, spanning tree, ...

6. Graph representations: adjacency list, adjacency matrix, incidence matrix.

7. Data structures:
   - FIFO queue and LIFO stack – enqueue, dequeue
   - Priority queue – Insert(S,u), Extract-Min(S), Decrease-Key(S,u,k), ...
   - Disjoint-set – Make-set(u), Union(u,v), Find-set(u)