Homework 9:
Greedy Algorithm:
1) State one example of Greedy algorithm other than that studied in class. Explain the algorithm in brief.

- Example: An example of a greedy algorithm is the fractional knapsack problem which optimizes the amount of items which can be put inside a bag of limited capacity by inserting the item with the highest value to size ratio at each step.
- Description: A greedy algorithm is an algorithm which can be used to find the shortest path between two vertices by making the optimal (shortest path) decision at each local step in the path in the hopes of resulting in a globally optimal solution. It calculates the shortest/most optimal path from the current node to each adjacent node at each step in order to accomplish this, until a relatively optimal path between the two nodes is completed.

Homework 10:
Dynamic Programming: Dynamic Programming is mainly an optimization over plain recursion. Wherever we see a recursive solution that has repeated calls for the same inputs, we can optimize it using Dynamic Programming. The idea is to simply store the results of subproblems, so that we do not have to re-compute them when needed later. This simple optimization reduces time complexities from exponential to polynomial.
Explain Dynamic Programming with one example other than the one studied in class.

- One example of a problem whose solution can be optimized via the usage of dynamic programming is the rod cutting problem. The rod cutting problem is a common interview question which states: “Given a rod of length n and a list of rod prices of length i, where 1 <= i <= n, find the optimal way to cut the rod into smaller rods to maximize profit”. This can be solved by the following recursive relation “rodcut(n) = max { price[i – 1] + rodCut(n – i) } where 1 <= i <= n”, but you can see that this will result in overlapping subproblems which will be computed multiple times unnecessarily. By memoizing the solutions to smaller subproblems and storing them in a map rather than computing them over and over again, we can reduce the time and space complexities of this problem.

Homework 11:
Concurrency:
Explain the following briefly with the help of an example:
a) Race around condition

- A race around condition occurs when a computer performs multiple operations simultaneously which must be performed in a specific order to get the desired result. This can happen when multiple computer threads try to access the same memory location at the same time.
- One example of the race around condition is the Therac-25 incident in which a chemotherapy machine had race condition errors which resulted in the machine

outputting inappropriately high doses of radiation, leading to deaths of the patients.

b) Critical Section

- Critical sections are parts of the code executed by multiple threads, which also access the same memory locations
- Example: if thread a and thread b both print out the value of a variable a, then change the value of the variable, the order of the values that will be printed out cannot be predetermined.

c) Mutual Exclusion

- Mutual exclusion is a technique used to prevent race condition errors in critical sections of code by ensuring that if one process is executing in a critical section, no other process can be executing in that section at the same time to ensure that shared resources remain synchronized and consistent.
- Example: in the previous example, if we use the principle of mutual exclusion, we can make sure that the values are printed in a predetermined order every time.