1 Define Dynamic Programming and explain with different examples.

Dynamic Programming is both a mathematical optimization method and a computer programming method. It is a methodology that often yield polynomial time algorithms; it solves problems by combining the results of solved overlapping sub problems.
Dynamic Programming is mainly an optimization over plain recursion. Wherever we see a recursive solution that has repeated calls for same inputs, we can optimize it using Dynamic Programming.
One of the examples is **Fibonacci numbers**, they are the numbers in the following integer sequence. 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ... If we write simple **Recursive solution** for Fibonacci Numbers, we can observe that this implementation does a lot of repeated work and we get exponential time complexity.
But We can avoid the repeated work done if we optimize it by storing solutions of sub problems by using **Dynamic Programming**. By doing this, time complexity reduces to linear.
Another example is **Ugly Numbers**, they are numbers whose only prime factors are 2, 3 or 5. The sequence 1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, ... shows the first 11 ugly numbers. By convention, 1 is included.
To find the ugly number we can run a **Loop** for all positive integers until ugly number count is smaller than n, if an integer is ugly than increment ugly number count. But this method is not time efficient as it checks for all integers until ugly number count becomes n, but space complexity of this method is O (1). If we use **Dynamic Programming**, there will be a time efficient solution with O(n) extra space. The ugly-number sequence is 1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, ... because every number can only be divided by 2, 3, 5. We can find that every subsequence is the ugly-sequence itself (1, 2, 3, 4, 5, ...) multiply 2, 3, 5. Then we use similar merge method as merge sort, to get every ugly number from the three subsequence.