```
class Knapsack {
    // Function to solve the 0-1 Knapsack problem using DP
    static int knapsack(int W, int[] wt, int[] val, int n) {
        int[][] dp = new int[n + 1][W + 1]; // DP table banate hain

        for (int i = 0; i <= n; i++) {
            for (int w = 0; w <= W; w++) {
                if (i == 0 || w == 0) {
                    dp[i][w] = 0; // Agar koi item ya weight na ho toh value 0 hogi
                } else if (wt[i - 1] <= w) {
                    dp[i][w] = Math.max(val[i - 1] + dp[i - 1][w - wt[i - 1]], dp[i
- 1][w]);
                } else {
                    dp[i][w] = dp[i - 1][w];
                }
            }
        }
        return dp[n][W]; // DP table mein answer last cell mein hoga
    }

    public static void main(String[] args) {
        int[] val = { 60, 100, 120 }; // Item values
        int[] wt = { 10, 20, 30 };    // Item weights
        int W = 50;                    // Max knapsack capacity
        int n = val.length;            // Number of items

        System.out.println("Maximum value: " + knapsack(W, wt, val, n));
    }
}
```