Code :

```
import java.util.Scanner;

public class Knapsack01DP {

    // Function to solve 0-1 Knapsack using Dynamic Programming
    static int knapSack(int capacity, int[] weight, int[] value, int n) {
        int[][] dp = new int[n + 1][capacity + 1];

        // Build table dp[][] in bottom-up manner
        for (int i = 0; i <= n; i++) {
            for (int w = 0; w <= capacity; w++) {
                if (i == 0 || w == 0)
                    dp[i][w] = 0;
                else if (weight[i - 1] <= w)
                    dp[i][w] = Math.max(value[i - 1] + dp[i - 1][w - weight[i - 1]], dp[i - 1][w]);
                else
                    dp[i][w] = dp[i - 1][w];
            }
        }

        return dp[n][capacity];
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of items: ");
        int n = sc.nextInt();

        int[] value = new int[n];
        int[] weight = new int[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter value of item " + (i + 1) + ": ");
            value[i] = sc.nextInt();
            System.out.print("Enter weight of item " + (i + 1) + ": ");
            weight[i] = sc.nextInt();
        }

        System.out.print("Enter capacity of knapsack: ");
        int capacity = sc.nextInt();
        sc.close();

        int maxValue = knapSack(capacity, weight, value, n);
        System.out.println("\nMaximum value in knapsack = " + maxValue);
    }
}
```

Output :

```
Enter number of items: 3

Enter value of item 1: 60

Enter weight of item 1: 10

Enter value of item 2: 80

Enter weight of item 2: 20

Enter value of item 3: 100

Enter weight of item 3: 30

Enter capacity of knapsack: 40


Maximum value in knapsack = 160
```