

```
int knapsack(int[] w, int[] p, int n, int C)
```

```
    // w: vettore dei pesi
```

```
    // p: vettore dei profitti
```

```
    // n: numero di oggetti
```

```
    // C: capacità massima dello zaino
```

```
    // creo la tabella di programmazione dinamica
```

```
    DP ← new int[0...n][0...C]
```

```
    // la inizializzo
```

```
    from i ← 0 until n do
```

```
        | DP[i][0] = 0 // capacità nulla
```

```
    from c ← 0 until C do
```

```
        | DP[0][c] = 0 // nessun oggetto
```

```
    // calcolo caso per caso
```

```
    from i ← 1 until n do
```

```
        from c ← 1 until C do
```

```
            if  $w[i] \leq c$  then // se la capacità residua è sufficiente
```

```
                | DP[i][c] = max(DP[i - 1][c - w[i]] + p[i], DP[i - 1][c]) // deciso se prenderlo  
                | o scartarlo
```

```
            else
```

```
                | DP[i][c] = DP[i - 1][c] // lo scarto
```

```
    // restituisco
```