Hopscotch

This algorithm uses DP to store the maximum possible sum for each index on the hopscotch line by comparing the max sum of the previous square, the weight of the current square plus the max weight of the squares 2 and 3 jumps behind, respectively. Then the max sum of that index is chosen by picking the max of the three choices listed.

Pseudocode

```
hops = inputArray
let dpArray = [n]
dpArray[1]=hops[1]
dpArray[2]=hops[2]
dpArray[3]=hops[1]+hops[3]
for i in range (4,n):
    dpArray[i] = max(dpArray[i-1],dpArray[i-2]+hops[i],dpArray[i-3]+hops[i])
return dpArray[n]
```

The first three are manually assigned so there is no risk of referencing a negative index

Running time:

This algorithm runs in O(n) time because it only needs one run through of the array to determine the max sum.