Algorithms Homework 3

Name : Fatih Goncagül

Recursive Algorithm

```
public  static  int recursiveSol(int n){

    if (n<1){
        return 0;
    }else if (n>=5  && n!=8 && n!=12  ){
        return 1 + recursiveSol(n - 5);
    }else if (n>=4 ){
        return 1 + recursiveSol(n-4);
    }else {
        return 1+ recursiveSol(n-1);
    }

}
```

With the first " if " statement my algorithm does not always choose the maxiumum weighted element, cause chosing a max weighted element all the time does not mean that we will get smallest number of items.

Dynamic Programming Algorithm

We are told to pick whatever subset of these 3 items with weights : 1 kg, 4 kgs, 5kgs.The main focus of dynamic programming is subproblems and knowing the solution to a subproblem so that we can use this to optimize my solution to my problem based on previous solutions.Since thief wants to steal smallest number of items, we need to maximize the weight when we make a choise cause only that way we will get smallest number of items.Of course chosing max weighted element all the time does not mean that we will get smallest number of items.

```
public  static int dynamicProg( int n){

    int[] array = new int[n+1];
    array[0] = 0;

    for (int i = 0;i<=n ;i++){
        if (i<4){
            array[i] = i;
        }else if (i<5){
            array[i] = Math.min(array[i - 4], array[i - 1]) +1;
        }
        else {
                array[i] = Math.min(array[i-5],Math.min(array[i - 4],
array[i - 1]) ) +1;
            }
        }

return array[n];

}
```

Mathematical recurrence relation :

We will have basically two choices

- Steal an item
- Do not steal an item

Of course these choices will be made with considering our goal which is smallest number of items with a weight capacity.

n is the weight capacity of the bag.
And mathematical recurrence relation would be :

n : it n<4

C(n) = min(c(min((n-4),c(n-1)) + 1 : if n is greater than or equal to 4 and n<5

min((n-5), min(C(n-4), C(n-1))) + 1 : for the otherwise

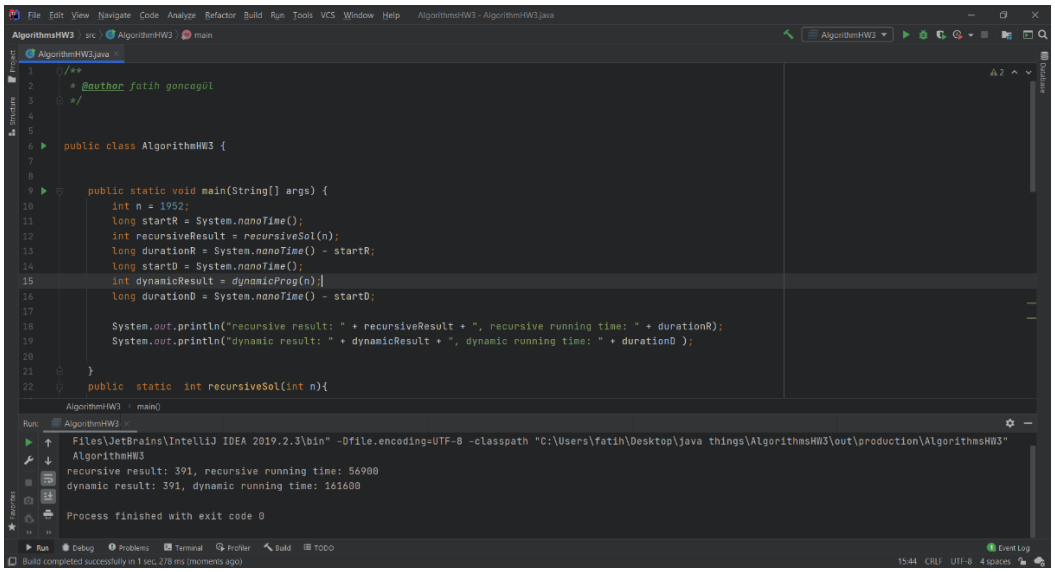
Comparison of recursive algorithm and dynamic programming algorithm

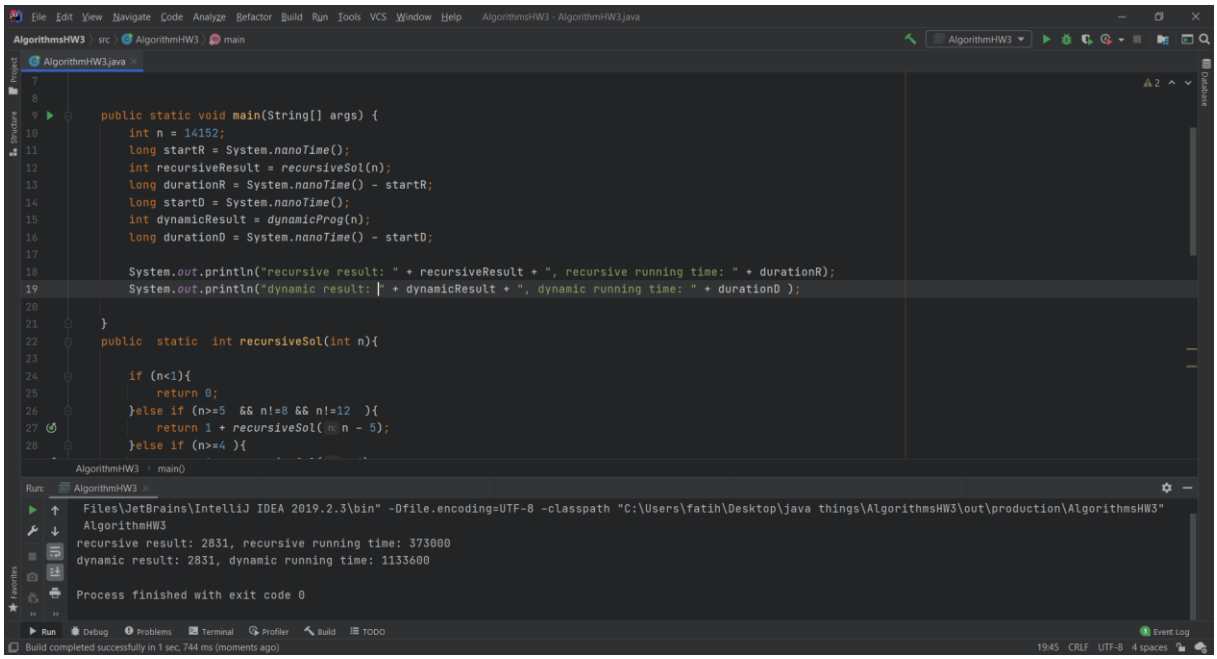
Here I have the running time of the algorithms with different bag capasities.

Capacity of bag (n)	Recursive Algorithm	Dynamic Prog. Algorithm
10	1 800	24 200
100	2 400	42 800
1 000	25 000	116 800
10 000	280 400	739 200

The reason why dynamic programming algorithm is a little bit slower is that it deals with redundant subproblems.The time complexity of dynamic programming algorithms would be O(n) because I have only one for loop there.ss

Here are two random numbers that I have tried and both algorithms finds the smallest number of items.




File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AlgorithmsHW3 - AlgorithmHW3.java
AlgorithmsHW3 › src › AlgorithmHW3 › main
AlgorithmHW3
AlgorithmHW3.java
public static void main(String[] args) {
int n = 14152;
long startR = System.nanoTime();
int recursiveResult = recursiveSol(n);
long durationR = System.nanoTime() - startR;
long startD = System.nanoTime();
int dynamicResult = dynamicProg(n);
long durationD = System.nanoTime() - startD;
System.out.println("recursive result: " + recursiveResult + ", recursive running time: " + durationR);
System.out.println("dynamic result: " + dynamicResult + ", dynamic running time: " + durationD);
}
public static int recursiveSol(int n){
if (n<1){
return 0;
}else if (n>=5 && n!=8 && n!=12){
return 1 + recursiveSol(n - 5);
}else if (n>=4){
AlgorithmHW3 › main()
Run: AlgorithmHW3
Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.encoding=UTF-8 -classpath "C:\Users\fatih\Desktop\java things\AlgorithmsHW3\out\production\AlgorithmsHW3"
AlgorithmHW3
recursive result: 2831, recursive running time: 373000
dynamic result: 2831, dynamic running time: 1133600
Process finished with exit code 0
Run Debug Problems Terminal Profiler Build TODO
Event Log
Build completed successfully in 1 sec, 744 ms (moments ago)
19:45 CRLF UTF-8 4 spaces