	Student information	Date	Number of session
Algorithmics	UO: 271407	09/03/2020	3
	Surname: Llera		
	Name: Elías		

Escuela de Ingeniería Informática
Universidad de Oviedo

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Activity 1. Basic recursive methods.

Subtraction4:

```
public static long rec4(int n) {
		long cont = 0;
		if (n<=0)
			cont++;
		else {
			cont++;      // O(1)
			rec4(n-2);
			rec4(n-2);
			rec4(n-2);
		}
		return cont;
	}
```

This method follows a divide and conquer approach, specifically by subtraction. We can see that the number of subproblems is 3, and hence a=3. Also, each subproblem is two units smaller than the parent problem, so b=2. Furthermore, the complexity of the non-recursive part of the algorithm is O(1) (the only instruction is to increase a counter, which is constant). Taking all of this into consideration, we can see that a>1 and then the complexity is O($a^{n/b}$). With the numbers we have already explained, we can see that the result is O($3^{n/2}$).

Algorithmics	Student information	Date	Number of session
	UO: 271407	09/03/2020	3
	Surname: Llera		
	Name: Elías		

Division4:

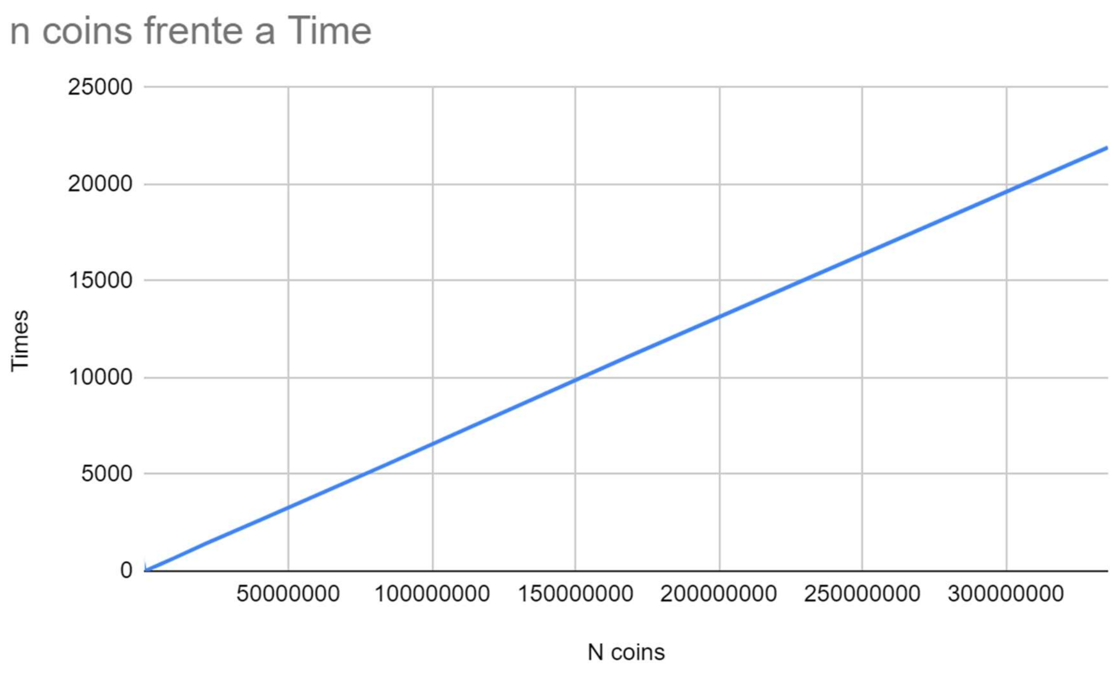
```
public static long rec4 (int n) {
            long cont = 0;
            if (n<=0)
                  cont++;
            else {
                  cont++;      // O(1)
                  rec4(n/2);
                  rec4(n/2);
                  rec4(n/2);
                  rec4(n/2);
            }
            return cont;
      }
```

This method follows a divide and conquer approach, specifically by division. We can see that the number of subproblems is 4, and hence a=4. Also, each subproblem has a size half of its parent's size, so b=2. Furthermore, the complexity of the non-recursive part of the algorithm is O(1) (the only instruction is to increase a counter, which is constant). Taking all of this into consideration, we can see that $a>b^k$ ($4>2^0$) and then the complexity is $O(n^{\log_b a})$ With the numbers we have already explained, we can see that the result is $O(n^{\log_2 4})$ = $O(n^2)$.

Algorithmics	Student information	Date	Number of session
	UO: 271407	09/03/2020	3
	Surname: Llera		
	Name: Elías		

Activity 2. Uncle Scrooge.

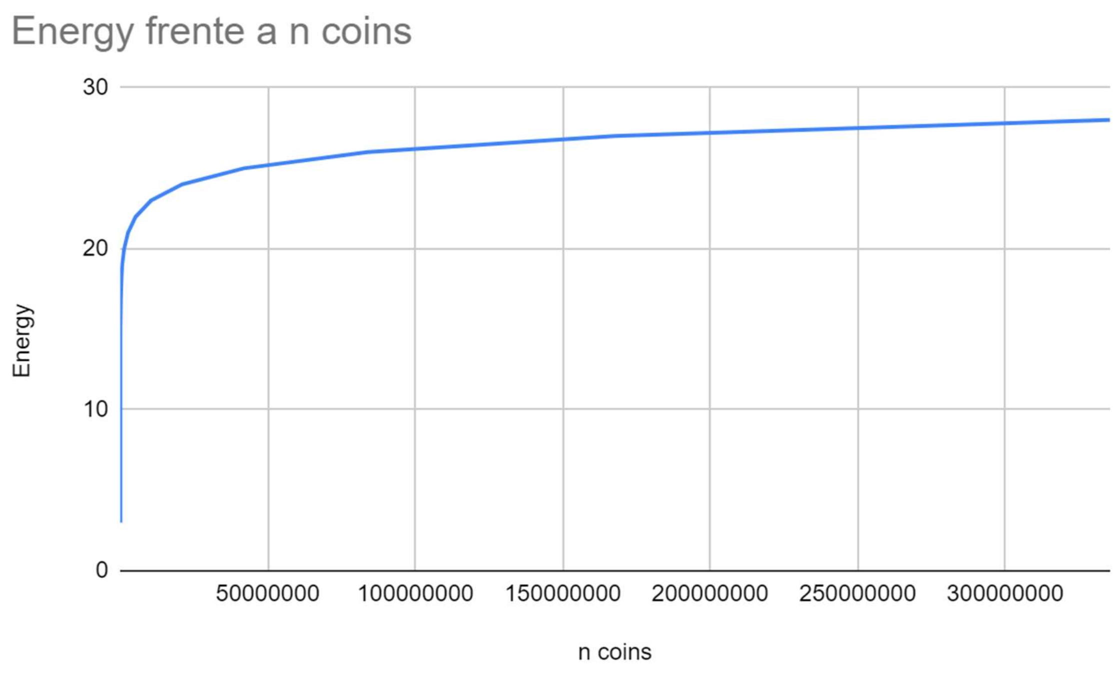
N coins against time



In this graph, we can see the time relation between the number of coins (the workload) and the time taken to execute the algorithm. If we have a look at the algorithm, we can see that it follows a divide and conquer by division approach. Furthermore, we can see that a=1, b=2 and k=1. According to the formula, when $a<b^k$ the time complexity is $O(n^k)$. We can see that in this case, the mentioned condition is met, and then the complexity is O(n), which makes sense with the results obtained and showed in the graph.

Algorithmics	Student information	Date	Number of session
	UO: 271407	09/03/2020	3
	Surname: Llera		
	Name: Elías		

Energy against n coins



In the graph, we can see that the energy taken to measure the coins follows a logarithmic trend. This curve makes sense with the values obtained from the measures, since after doubling the number of coins (the workload), the cost of energy only incremented in one unit. In the graph we can see how the cost drastically increases at the beginning, but starts to approach a constant value for big numbers of coins.

CPU: Intel® Core™ i5-3470 CPU @ 3.20GHz

RAM: 8GB