


| Algorithmics | Student information | Date | Number of session |
|--------------|-----------------------|---|-------------------|
| | UO: 277921 | 2/3/2021 | 3.1 |
| | Surname: García López |  Escuela de Ingeniería Informática Universidad de Oviedo | |
| | Name: Rosa | | |



Activity 1. Basic recursive models

Firstly, I want to remark that in all the classes; even the new ones, if the integer passed as a parameter is less than or equal to zero, the complexity would be $O(1)$ because of a condition on the function. Having that in mind, I would explain the cases when the integer is greater than zero.

- **Substraction1:** On this class the number of subproblems ' a ' is 1, the constant we subtract by, ' b ', is 1 as well and the overall complexity excluding the recursive calls is $O(n^k) = O(1)$; therefore, $k = 0$. Now, we know all the parameters and, because $a = 1$ we apply the formula $O(n^{k+1})$ to calculate the complexity; that is, $O(n)$.
- **Substraction2:** On this class the number of subproblems ' a ' is 1, the constant we subtract by, ' b ', is 1 as well and the overall complexity excluding the recursive calls is $O(n^k) = O(n)$; therefore, $k = 1$. Now, we know all the parameters and, because $a = 1$ we apply the formula $O(n^{k+1})$ to calculate the complexity; that is, $O(n^2)$.
- **Substraction3:** On this class the number of subproblems ' a ' is 2, the constant we subtract by, ' b ', is 1 and the overall complexity excluding the recursive calls is $O(n^k) = O(1)$; therefore, $k = 0$. Now, we know all the parameters and, because $a > 1$ we apply the formula $O(a^{n/b})$ to calculate the complexity; that is, $O(2^n)$.
- **Division1:** On this class the number of subproblems ' a ' is 1, the constant we subtract by, ' b ', is 3 and the overall complexity excluding the recursive calls is $O(n^k) = O(n)$; therefore, $k = 1$. Now, we know all the parameters and, because $a < b^k$ we apply the formula $O(n^k)$ to calculate the complexity; that is, $O(n)$.
- **Division2:** On this class the number of subproblems ' a ' is 2, the constant we subtract by, ' b ', is 2 as well and the overall complexity excluding the recursive calls is $O(n^k) = O(n)$; therefore, $k = 1$. Now, we know all the parameters and, because $a = b^k$ we apply the formula $O(n^k * \log n)$ to calculate the complexity; that is, $O(n \log n)$.

| Algorithmics | Student information | Date | Number of session |
|--------------|-----------------------|----------|-------------------|
| | UO: 277921 | 2/3/2021 | 3.1 |
| | Surname: García López | | |
| | Name: Rosa | | |

- **Division3:** On this class the number of subproblems ' a ' is 2, the constant we subtract by, ' b ', is 2 as well and the overall complexity excluding the recursive calls is $O(n^k) = O(1)$; therefore, $k = 0$. Now, we know all the parameters and, because $a > b^k$ we apply the formula $O(n^{\log_b a})$ to calculate the complexity; that is, $O(n)$.
- **Substraction4:** On this case the complexity is known, $O(3^{n/2})$ and it is divide and conquer by subtraction so we have two possibilities; the number of subproblems ' a ' is equal to one or it is greater than one. If we have a look at the complexity we can notice that the value of a is 3 and the value of b is 2; because we can only have that type of complexity applying the formula $O(a^{n/b})$. The value of k is not relevant here, but in my case it is 0.
- **Division4:** On this case the complexity is known, $O(n^2)$. If we have a look at the complexity we can notice that the value of a is either greater than b^k or less than b^k ; because the complexity is not of the form $O(n^k * \log n)$. I have chosen the case on which $a > b^k$, so the formula to calculate the complexity we apply is $O(n^{\log_b a})$. Having in mind the previous premises, we can conclude that the value of a is 4 and the value of b is 2 to obtain the said complexity. Again, the value of k is not relevant, but in this case it is 0.