Algorithmics	Student information	Date	Number of session
	UO: 284185	1/3/2023	3
	Surname: Fernández-Catuxo Ortiz		
	Name: Rita		

Escuela de Ingeniería Informática
Universidad de Oviedo

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Activity 1. Basic recursive models.

DIVISION		SUBTRACTION	
$O(n^k)$	if $a < b^k$	$O(n^k)$	if $a < 1$
$O(n^k * \log n)$	if $a = b^k$	$O(n^{k+1})$	if $a = 1$
$O(n^{\log_b a})$	if $a > b^k$	$O(a^{n/b})$	if $a > 1$

- **A brief explanation for each of the given classes indicating how you calculated the complexity of that class**

1. SUBTRACTION1

a = 1, b = 1, k = 0

a = 1 because we only make one recursive call. b = 1 because we subtract one in the call, and k = 0 because excluding recursive calls, the complexity of the method is O (1). Then, as we the recursive call is a subtraction, we are going to use the scheme by subtraction and thus, as a = 1, we have a complexity of O (n^k+1) = O (n).

2. SUBTRACTION2

a = 1, b = 1, k = 1

a = 1 because we only make one recursive call. b = 1 because we subtract one in the call, and k = 1 because excluding recursive calls, the complexity of the method is O (n). Then, as we the recursive call is a subtraction, we are going to use the scheme by subtraction and thus, as a = 1, we have a complexity of O (n^k+1) = O (n^2).

Algorithmics	Student information	Date	Number of session
	UO: 284185	1/3/2023	3
	Surname: Fernández-Catuxo Ortiz		
	Name: Rita		

3. SUBTRACTION3

a = 2, b = 1, k = 0

a = 1 because we make two recursive call. b = 1 because we subtract one in the call, and k = 0 because excluding recursive calls, the complexity of the method is O (1). Then, as we the recursive call is a subtraction, we are going to use the scheme by subtraction and thus, as a = 1, we have a complexity of O (a^(n/b)) = O(2^n).

4. DIVISION1

a = 1, b = 3, k = 1

a = 1 because we make two recursive call. b = 3 because we divide by three in the call, and k = 1 because excluding recursive calls, the complexity of the method is O (n). Then, as we the recursive call is a division, we are going to use the scheme by division and thus, as 1 < 3^1 (a < b^k), we have a complexity of O (n^k) = O (n).

5. DIVISION2

a = 2, b = 2, k = 1

a = 2 because we make two recursive calls. b = 2 because we divide by two in the call, and k = 1 because excluding recursive calls, the complexity of the method is O (n). Then, as we the recursive call is a division, we are going to use the scheme by division and thus, as 2 = 2^1 (a = b^k), we have a complexity of O (n * log n).

Algorithmics	Student information	Date	Number of session
	UO: 284185	1/3/2023	3
	Surname: Fernández-Catuxo Ortiz		
	Name: Rita		

6. DIVISION3

a = 2, b = 2, k = 0

a = 2 because we make two recursive calls. b = 2 because we divide by two in the call, and k = 0 because excluding recursive calls, the complexity of the method is O (1). Then, as we the recursive call is a division, we are going to use the scheme by division and thus, as 2 > 2^0 (a > b^k), we have a complexity of O (n).

- **A brief explanation for each of the 2 new classes indicating how you calculate the complexity to get the requested one.**

1. SUBSTRATION4

a = 3, b = 2, k = 0

a = 3 because we make two recursive calls. b = 2 because we subtract two in the call, and k = 0 because excluding recursive calls, the complexity of the method is O (1). Then, as we the recursive call is a subtraction, we are going to use the scheme by subtraction and thus, as 3 > 1 (a > 1), we have a complexity of O (3 ^ (n / 2)).

Algorithmics	Student information	Date	Number of session
	UO: 284185	1/3/2023	3
	Surname: Fernández-Catuxo Ortiz		
	Name: Rita		

2. DIVISION4

a = 4, b = 2, k = 1

a = 4 because we make two recursive calls. b = 2 because we divide by two in the call, and k = 1 because excluding recursive calls, the complexity of the method is O (n). Then, as we the recursive call is a division, we are going to use the scheme by division and thus, as 4 > 2^1 (a > b^k), we have a complexity of O (n^2).