TD 4 DEV5.1 Yanis DARIUS

EX2

function1 =O(n*m)
avec n qui représente le nombre d'élément dans le tableau 1 et m qui représente le nombre d'élément dans le tableau 2 pour chaque élément du tableau n on vas parcourir a chaque fois m donc n*m

function2 = O(n)
la complexité dépend de la variable x donc O(n)

function3 = O(1)
il n y a pas itération dans le code

EX3

bubblesort = pire cas O(n^2) / meilleure cas O(n)
car il y a deux boucle pour triée le tableau

find_rank_student = O(n^2) car bubblesort a un ordre de grandeur supérieur

sort_students = O(m*n^3)
avec n = student number et m = grade number
n^3 = car find rank bloucle sur student number
m = car la première boucle littéraire sur grade number

EX4

insertion_sort = O(n^2)
c’est une fonction qui tri à double itération

total_sum = O(n)
il fait la somme sur tous les élément d' un tableau multidimensionnel
soit n tout les element du tableau multidimensionnel

sort_with_sum = O(n^2)
une fonction de tri classique qui tri en fonction de la somme du tableau total_sum n’est pas influent sur sort_with_sum puis que sa complexité n’est pas du même rang

TD 4 DEV5.1 Yanis DARIUS

sort_nd_array = O(n^d*n^2)
avec n le nombre d'élément à chaque niveau du tableau et d le nombre de dimension
n^d = dimension
n^2 = complexité de l algorithme de tri

l algorithme améliorer a une complexité de O(n^d*(n*log(n))