Written Assignment #2

For chapter 4

1. R-4.13 page 194: order the following functions by asymptotic growth rate:

|  |  |  |
| --- | --- | --- |
| 4nlogn + 2n | 210 | 2 logn |
| 3n + 100 log n | 4n | 2 n |
| n2 + 10 n | n3 | n log n |
|  |  |  |

1. R-4.16- 20 page 194-5 Give a big-Oh characterization, in terms of n, of the running time of the methods Ex1 to Ex4 , shown in Code Fragment 4.6 of page 195.

**1. R-4.15**

**Alg Ex1(A):**

Input: array A storing n > 0 integers.

Output : The sum of the elements in A.

**s <-- A[0]**

**for i <-- 1 to n-1 do**

**s <-- s + A[i]**

**return s**

**2. R-4.16**

**Alg Ex2(A):**

**Input: array A storing n > 0 integers.**

**Output : The sum of the elements at even cells in A.**

**s <-- A[0]**

**for i <-- 2 to n-1 by increments of 2 do**

**s <-- s + A[i]**

**return s**

**3.R-4.17**

**Alg Ex3(A) :**

**Input: array A storing n > 0 integers.**

**Output : The sum of the prefix sums in A.**

**s <-- 0**

**for i <-- 0 to n-1 do**

**s <-- s + A[0]**

**for j <-- 1 to i do**

**s <-- s + A[j]**

**return s**

**4.R-4.18**

**AlgEx4(A) :**

**Input: array A storing n > 0 integers.**

**Output : The sum of the prefix sums in A.**

**s <-- A[0]**

**t <-- s**

**for i <-- 1 to n-1 do**

**s <-- s + A[i]**

**t <-- t + s**

**return s**

**5.R-4.19**

**AlgEx5(A,B) :**

**Input: Arrays A and B each storing n > 0 integers.**

**Output : The number of elements in B equal to the sum of prefix sums in A.**

**c <-- 0**

**for i <-- 0 to n-1 do**

**s <-- 0**

**for j <-- 0 to n-1 do**

**s <-- s + A[0]**

**for k <-- 1 to j do**

**s <-- s + A[k]**

**if B[i] = s then**

**c <-- c + 1**

**return c**

1. R-4.22 page 194: For each function f(n) and time t in the following table, determine the largest size n of a problem P that can be solved in time t if the algorithm for solving P takes f(n) microseconds.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 second | I hour | 1 month | 1 century |
| *log n* | ≈ 10 300000 |  |  |  |
| n |  |  |  |  |
| *n log n* |  |  |  |  |
| n2 |  |  |  |  |
| 2n |  |  |  |  |

1. **C-4.9 page 198 Show that**  **is O(n3) .**