# DESIGN AND ANALYSIS OF ALGORITHMS

|  |  |
| --- | --- |
| **Name** | **Hồ Minh Cường** |
| **IRN** | **1331209040** |

# HOMEWORK 3

## Question 1

**Thread**: Compute the following sums:

**Answer:**

= +

= 0(0 + 1) + 1(1 + 1) + … + n(n + 1) + (n + 1)(n + 1 + 1) + (n + 2)(n + 2 + 1)

= 0.1 + 1.2 + 2.3 + … + n(n + 1) + (n + 1)(n + 2) + (n + 2)(n + 3)

= + (n + 1)(n + 2) + (n + 2)(n + 3)

=

=

= 32

= 32 (31 + 32 + 33 + … + 3n )

= 32

=

=

= =

=

## Question 2

**Thread**: Consider the following algorithm:

**ALGORITHM**

**function** Secret(A[0…*n* - 1])

* Input: An array A[0…*n* - 1] of n real numbers

minval 🡨 A[0]; maxval 🡨 A[0]

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

**if** A[*i*] < *minval* **then**

*minval* 🡨 A[*i*]

**if** A[*i*] > maxval **then**

*maxval* 🡨 A[*i*]

**return** *maxval* *- minval*

(a) What does this algorithm do?

(b) What is its basic operation?

(c) How many times is the basic operation executed in the best and worst cases?

(d) What is the efficiency class of this algorithm?

**Answer**:

1. This algorithm is performed to find the maximum and minimum value in an array of real numbers.
2. Algorithm’s basic operations are substraction ( - ), greater than ( > ) and less then ( < ).
3. The basic operation executed in the best and worst cases is (n - 1) times.
4. The efficiency class of this algorithm is (n).

## Question 3

**Thread:** Solve the following recurrence relations, using backwards substitution, or by calculating the first few terms and generalizing.

1. x(n) = 4 x(n - 1) for n > 1; x(1) = 2
2. x(n) = x(n - 1) + n for n > 0; x(0) = 3
3. x(n) = x(n/3) + 1 for n > 1; x(1) = 1 (solve for n = 3i)

**Answer:**

1. x(n) = 4 x(n - 1)

= 4i x(n - i)

With (n – i) = 1 => i = n – 1

x(n) = 4(n - 1) x(1)

= 23.4n

= O(4n).

1. x(n) = x(n - 1) + n

= x(n - i) + 3i

With (n - i) = 0 => i = n

x(n) = x(0) + 3n

= 3 + 3n

= O(n).

1. x(n) = x() + 1

= x() + i

With = 1 => n = 3i => i = log3(n)

x(n) = x(1) + log3(n)

= 1 + log3(n)

= O(log3(n))