Algorithms Design and Analysis

Computing Running time

Mahmoud Khalaf Saeed

Section (4)

Example 01

$$\text{for } (\mathrm{i} = 1\, ; \mathrm{i} \leq \mathrm{n}\, , \mathrm{i} + +)$$

$$\text{for } (\mathrm{j} = 1\, ; \mathrm{j} \leq 10\, , \mathrm{j} + +)$$

$$if\ (condition)$$

$$operation$$

$$else$$

$$break$$

$T(n) = \mathrm{O(n)} * \mathrm{O}\ (10) = \mathrm{O}\ (\mathrm{N})$

Example 02

$$for\ (i = 1\ ; i \leq n\ , i++)$$

$$for\ (j = 1\ ; j \leq n * n\ , j++)$$

$$operation$$

$$T(n) = O(\mathrm{n}) * O\ (\mathrm{n} * \mathrm{n}) = O\ (\mathrm{N}^3)$$

Example 03

$$\text{for } (\mathrm{i} = 1\,; \mathrm{i} \le \mathrm{n}\,, \mathrm{i} + +)$$

$$\text{for } (\mathrm{j} = \mathrm{i}\,; \mathrm{j} \le \mathrm{i} + 3\,, \mathrm{j} + +)$$

$$operation$$

$$\boldsymbol{T}(\boldsymbol{n}) = \boldsymbol{O}(\mathbf{n}) * \mathbf{O}\,(\mathbf{4}\,) = \mathbf{O}\,(\mathbf{N})$$

Note that inner loop executed from I to I + 3 only

Example 04

$$\text{for } (i = 1\,; i \le n\,, i++)$$

$$\text{for } (j = 1\,; j \le n\,, j += 5)$$

$$operation$$

$$\text{for } \left(k = 1\,; k \le \frac{n}{2}, k++\right)$$

$$operation$$

$$T(n) = MAX\,\{O(n) * O\,(n\,), O(n)\} = O\,(N^2)$$

Example 05

$$\text{for}\ (\mathrm{i} = 3\ ; \mathrm{i} \leq \mathrm{n}\ , \mathrm{i} + +)$$

$$operation$$

$$\boldsymbol{T(n) = \mathbf{O}(\mathbf{n})}$$

Example 06

$$\text{for}\ (\mathrm{i} = 3\ ; \mathrm{i} \leq \mathrm{n} * 2\ , \mathrm{i} + +)$$

$$operation$$

$$\boldsymbol{T(n) = \mathbf{O}(\mathbf{n})}$$

Example 07

$$\text{for } (i = 1\,; i \le n * m\,, i++)$$

$$operation$$

$$T(n) = \mathrm{O}(\mathrm{n} * \mathrm{m}) = O(N^2)\ if\ n \approx m$$

Example 08

$$for\ (i = 1\ ; i \le n\ , i = i * 2)$$
$$operation$$

Time complexity for loop means number of times the loop has run, the given loop will run for the following values of i :

i	1	2	4	8	16	……	…….	n
In terms of power of 2	2^0	2^1	2^2	2^3	2^4	…. ..	……	2^k

The loop will run till 2^k=n which leads to k = log N

$$T(n) = O(\log n)$$

Example 09

```
for i = 1 to N
    j = n
    while j > i
        j = j – 1
    end while
end for
```

Outer loop (i)	Inner loop (j)	# Steps that run
1	From n to 1	$n-1$
2	From n to 2	$n-2$
3	From n to 3	$n-3$
......	....	
n	From n to n	$n-n$

Summation of all steps =

$$\sum_{0}^{n-1} i = \frac{n-1\,(n-1+1)}{2} = \frac{n^2-n}{2} = O\,(n^2)$$

Example 10

$$\text{for } (\text{i} = 1 \text{ to k })$$
$$\text{for } (\text{x} = 1 \text{ to n })$$
$$x = x + 1$$
$$\text{for } (\text{m} = 1 \text{ to d})$$
$$m = m + 1$$

There are two inner loops , the first one

$$T1(n) = O(body) * NO\ of\ iteration = O(\text{n})$$

the second inner loop consume

$$T2(n) = O(body) * NO\ of\ iteration = O(\text{D})$$

The running time for outer loop is

$$T(n) = O(body) * NO\ of\ iteration$$
$$= O(\text{Max}[\text{N , D}]) * K = O\ (KN) where\ N\ is\ the\ \text{Max}[\text{N , D}]$$

Example 11

```
i = 1
    while i < N
        for (j = 1 to i )
            h = 1
        end for
        i = i * 2
    end while
```

Outer loop(i)	Inner loop(j)	# steps
$1 = 2^0$	From 1 to 1	$1 = 2^0$
$2 = 2^1$	From 1 to 2	$2 = 2^1$
$4 = 2^2$	From 1 to 4	$4 = 2^2$
$8 = 2^3$	From 1 to 8	$8 = 2^3$
......	....	.....
$= 2^k$	From 1 to 1	$x = 2^k$

The total number of steps =

$$2^0 + \mathbf{2^1} + \mathbf{2^2} + \mathbf{2^3} + \mathbf{2^4} + \ldots\ldots + \mathbf{2^n} = 1 + \sum_1^k 2^i$$

$$= 1 + 2^{logn+1} - 1 = 2^{logn+1} = 2 * 2^{logn} = 2\,n^{log2} = 2N$$

$$=> T(N) = O\,(N)$$

Outer loop terminates if i = n => 2^k = N =>k = log n

Example 12

```
i = n
    while i >  0
        i  = i − 10
    end while
```

$the\ i\ values\ decrease\ from$

$n\ by\ 10\ every\ time\ as:$

$N\ , N-10\ , N-20\ , N-\ 30\ ,\ \dots\dots\dots\ , N-\ H$

$which\ can\ be\ expressed\ as$

$$N-0*10\ , N-1*10\ , N-2*10\ , N-\ 3*10\ ,\ ..\ , N-k*10$$

The loop terminates when the condition is broken after N iteration with value equal to 0 so

$$N-k*10=0 \Rightarrow N=10\,k \Rightarrow k=\ n/10$$

$so\ while\ has\ n/10\ \ iterations\ which\ leads\ to\ T\ (N\ )=O(1)*\ n/10\ =O\ (n/10)\ \approx O\ (N)$

Input size: n = 10

Input size: n = 1 billion

Linear search takes 1 billion ms

Binary Search takes 32 ms

46 Days

Piece of Day

$T(N) = O(N)$

$T(N) = O(\log N)$

Assignment

Compute the running time of the following pseudocodes

```
for (int i = 1; i <= n; i += 2):
    print(i);                    ------------------------
```

```
for (int i = 1; i <= n; i += 20):
    print(i);                    ------------------------
```

```
for (int i = 0; i < n; i++):
    for (int j = 0; j < i; j++):
        print(j);
```

```
p = 0
for (int i = 1; p <= n; i += p):
    // statement;
```

```
for (int i = n; i >= 1; i /= 2):
    statement;
```

```
for (int i = 0; i*i < n; i++):
    // statement;
```

```
int fn(n):
    int a = n * n;
    print(a);
    return a;
```

Assignment instructions

Answer about all assignments as a word/powerPoint files .

Deadline of all assignments before next section

Assignments done by only one person

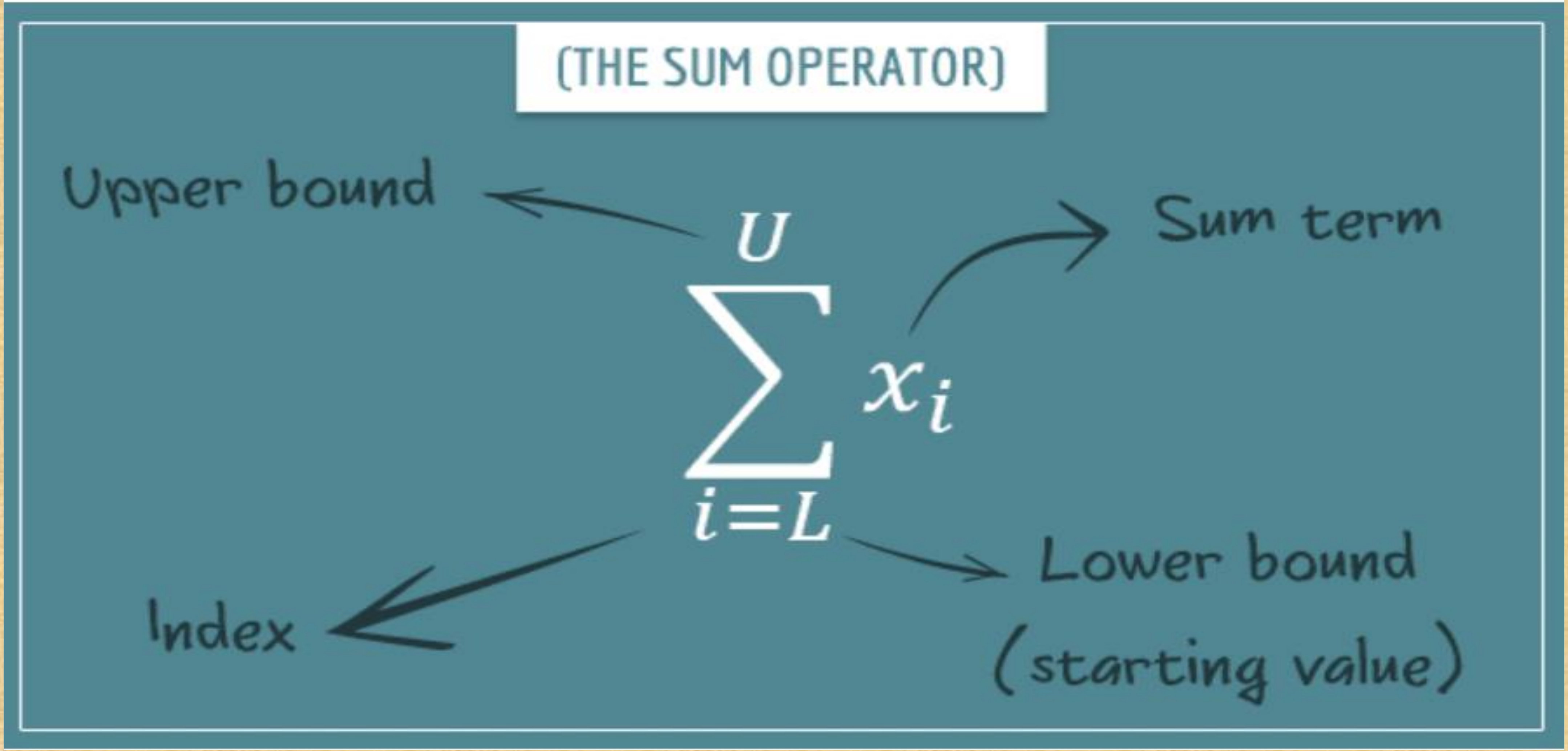
Similar assignments will not be checked (😏 BigZero 😒)

Deliver your assignment with your name on :

mahmoudelsaeed777@gmail.com

Math is needed everywhere

Summation Notation



Summation Notation

$\sum_{i=i_0}^{n} ca_i = c\sum_{i=i_0}^{n} a_i$ where c is any number. So, we can factor constants out of a summation.

$\sum_{i=i_0}^{n} (a_i \pm b_i) = \sum_{i=i_0}^{n} a_i \pm \sum_{i=i_0}^{n} b_i$ So, we can break up a summation across a sum or difference.

Summation Notation

$$\sum_{i=1}^{n} c = cn$$

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=1}^{n} i^3 = \left[\frac{n(n+1)}{2}\right]^2$$

Summation Notation

$$\sum_{i=1}^{\log n} 2^i = 2^{logn+1} - 1$$

Have a Good Day 😍