

Algorithm Complexity I

Exercise

ivanovitch.silva@ufrn.br
@ivanovitchm



```
def int_sum(n):
    sum = 0          #C1, 1
    while n > 0:     #C2, N
        sum += n     #C3, N
        n -= 1       #C4, N
    return sum       #C5, 1
```

```
int_sum(2)
```

```
3
```

```
int_sum(3)
```

```
6
```

```
int_sum(4)
```

```
10
```

Example #01

$$C_1 + N(C_2 + C_3 + C_4) + C_5$$

$$C_1 + C_5 + N(C_2 + C_3 + C_4)$$

$$\alpha + \beta N \gg O(N)$$

Example #02

```
def int_sum(n):  
    return n*(n+1)/2  #C1,C2,C3, 1
```

```
int_sum2(2)
```

```
3.0
```

```
int_sum2(3)
```

```
6.0
```

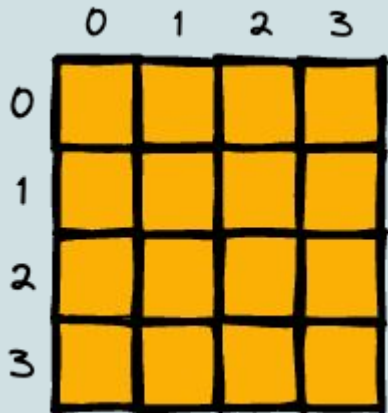
```
int_sum2(4)
```

```
10.0
```

$$C_1 + C_2 + C_3$$
$$\alpha \gg O(1)$$

Example #03

```
def print_matrix(mat):  
    for row in mat:           #C1, N  
        for cell in row:     #C2, N*N  
            print(cell)      #C3, N*N
```



$$\begin{aligned} &C_1N + C_2N^2 + C_3N^2 \\ &N^2(C_2 + C_3) + C_1N \\ &O(N^2) \end{aligned}$$

Example #04

```
def print_matrix(mat):  
    for row in mat:           #C1, N  
        for cell in row:     #C2, M*N  
            print(cell)      #C3, M*N
```

	0	1	2	3
0				
1				
2				
3				
4				

$$C_1N + C_2MN + C_3MN$$

$$MN(C_2 + C_3) + C_1N$$

$$O(MN)$$

Example #05

```
def foo(n):  
    for i in range(10000):    #C1, 10000  
        print(i)             #C2, 10000
```

$$C_1 10000 + C_2 10000$$

$$10000(C_1 + C_2)$$

$$O(1)$$

Example #06

```
def foo(n):  
    sum = 0 #C1, 1  
    for i in range(n): #C2, N  
        for j in range(2*n): #C3, 2N2  
            sum += j #C4, 2N2  
    for i in range(100*n): #C5, 100N  
        sum += i #C6, 100N  
    for i in range(4): #C7, 4  
        for j in range(3): #C8, 12  
            sum += j #C9, 12  
    return sum #C10, 1
```

$$4N^2 + 201N + 30$$

$$4N^2$$

$$O(N^2)$$

Example #07

```
def foo(n):  
    for i in range(1, n+1): #C1, n  
        for j in range(i): #C2, n*[i]  
            print(j)        #C3, n*[i]
```

How many times "print(j)" will be executed?

$$1 + 2 + 3 + 4 + \dots N$$

$$S_N = \frac{N(a_1 + a_N)}{2} \quad S_N = \frac{N^2}{2} + \frac{N}{2}$$

$$S_N = \frac{N(1+N)}{2} \quad O(N^2)$$

Example #08

```
def intersection(arr1, arr2):  
    for elem in arr1:           #C1, n  
        if elem in arr2:       #C2, n*m  
            print(elem)        #C3, n*m
```

$$(C_2 + C_3)NM + C_1N$$
$$O(NM)$$

$$O(N + M)$$

```
def both_contain(arr1, arr2, k):  
    return k in arr1 and k in arr2  #C1,n #C2,m
```

Example #09

```
def foo(n):  
    i = 1           #C1, 1  
    while i < n:    #C2, logn  
        i *= 2      #C3, logn
```

$$2 \log N + 1$$

$$O(\log N)$$

$$i = [1, 2, 4, 8, 16, \dots N]$$

$$i = [1, 2, 2*2, 2*2*2, 2*2*2*2, \dots N]$$

$$2^k = N \quad \log 2^k = \log N \quad k = \log N$$

```

def where_equal(str1, str2):
    indexes = []           #C1, 1
    i = 0                  #C2, 1
    while i < len(str1) and i < len(str2): #C3, min(n,m)
        if str1[i] == str2[i]:           #C4, min(n,m)
            indexes.append(i)             #C5, min(n,m)
        i += 1                            #C6, min(n,m)
    return indexes                  #C7, 1

```

$$4\min(n, m) + 3$$

$$O(\min(n, m))$$

Example #10