AYUDANTÍA 0: ESTRUCTURAS DE DATOS Y ALGORITMOS $2^{\underline{do}}$ Semestre 2020

Profesor: Javier Carrión
Ayudantes: Yerko Ortiz, Nicolás Nuñez

Objetivo de la ayudantía: Estudiar de cotas asintóticas.

Problema introductorio

Sea un número entero N, su tarea es implementar un algoritmo que calcula la suma de los primeros N naturales.

- ¿De cuántas formas posibles usted puede implementar este algoritmo?
- ¿Cuál es la forma más eficiente en tiempo, que se le ocurre?

Análisis de algoritmos

Para cada uno de los siguientes códigos, indique su complejidad en tiempo usando la notación Big-Oh $O(f(n))$. Además de realizar una breve descripción de cada algoritmo.

Algoritmo a

```
1  static int a_cont = 1;
2  static void a(int n) {
3      if(n == 0) {
4          System.out.println(a_cont++);
5          return;
6      }
7      for(int i = 1; i < n; ++i)
8          a(n - 1);
9      System.out.println(a_cont++);
10 }
```

Algoritmo b

```
1  static int b_cont = 0;
2  static void b(int k, int n) {
3      if (k == n)
4          System.out.println(b_cont++);
5      else {
6          b(k + 1, n);
7          b(k + 1, n);
8      }
9  }
```

Algoritmo c

```
1  static void c(int x) {
2          int n = 1;
3          while(n <= x){
4              System.out.println(n);
5              n *= 2 ;
6          }
7      }
```

Algoritmo d

```
1      static void d(int arr[], int n) {
2          for(int i = 0; i < n/8; ++i)
3              System.out.println(i);
4      }
```

"Becoming sufficiently familiar with something is a substitute for understanding it."

John H. Conway

Algoritmo e

```
1   static void e(int arr[], int n){
2       for(int i = 0; i < n; ++i)
3           System.out.println(i);
4   }
```

Algoritmo f

```
1   static void f(int arr[], int n, int k){
2       if (k >= n/3) return;
3       System.out.println(arr[k]);
4       f(arr, n, k + 1);
5   }
```

Algoritmo g

```
1   static void g(int n){
2       int cont = 0;
3       for(int i = 0; i < n; ++i){
4           for(int j = i; j < n; ++j){
5               for(int k = 1; k < n; k *= 2){
6                   System.out.println(cont);
7               }
8           }
9       }
10  }
```

Algoritmo h

```
1   static void h(int arr1[]) {
2       int n = arr1.length;
3       for (int i = 1; i < n; ++i) {
4           int key1 = arr1[i];
5           int j = i - 1;
6           while (j >= 0 && arr1[j] > key1) {
7               arr1[j + 1] = arr1[j];
8               j = j - 1;
9           }
10          arr1[j + 1] = key1;
11      }
12  }
```

Algoritmo i

```
1   static void i(int n){
2       for (int i = 1; i < n; i++){
3           for (int j = i + 1; j <= n; j++){
4               for (int k = 1; k <= j; k++){
5                   System.out.println(k);
6               }
7           }
8       }
9   }
```