|  |  |
| --- | --- |
|  | **Caratula para entrega de Prácticas** |
| Facultad de Ingeniería | Laboratorio de docencia |

Laboratorios de computación salas A y B

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                                Profesor:\_\_\_\_Ing. Marco Antonio Martinez Quintana\_\_\_

                            Asignatura:\_Estructura de Datos y Algoritmos I \_

                            Grupo:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_17\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                            No. de Práctica(s):\_\_\_\_\_\_\_\_\_\_\_11\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                            Integrante(s):Francisco Moises Barrera Guardia

                            No. de Equipo de  
                            cómputo empleado:\_\_\_\_\_\_\_\_\_\_13\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                            No. Lista o Brigada:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                            Semestre:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2°\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                            Fecha de entrega:\_\_\_\_\_\_\_Marzo del  2020\_\_\_\_\_\_\_\_\_

                            Observaciones:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                                                        \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                                                     CALIFICACIÓN:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Objetivo.

El objetivo de esta guía es implementar, al menos, dos enfoques de diseño (estrategias) de algoritmos y analizar las implicaciones de cada uno de ellos.

# Introducción.

## Fuerza bruta

El objetivo de resolver problemas por medio de fuerza es bruta es hacer una búsqueda exhaustiva de todas las posibilidades que lleven a la solución del problema. Un ejemplo de esto es encontrar una contraseña haciendo una combinación exhaustiva de caracteres alfanuméricos generando cadenas de cierta longitud. La desventaja de resolver problemas por medio de esta estrategia es el tiempo que toman.

## Algoritmos ávidos (greedy)

Esta estrategia se diferencia de fuerza bruta porque va tomando una serie de decisiones en un orden específico, una vez que se ha ejecutado esa decisión, ya no se vuelve a considerar. En comparación con fuerza bruta, ésta puede ser más rápida; aunque una desventaja es que la solución que se obtiene no siempre es la más óptima.

## Bottom-up (programación dinámica)

El objetivo de esta estrategia es resolver un problema a partir de subproblemas que ya han sido resueltos. La solución final se forma a partir de la combinación de una o más soluciones que se guardan en una tabla, ésta previene que se vuelvan a calcular las soluciones. Como ejemplo, se va a calcular el número n de la sucesión de Fibonacci.

La sucesión de Fibonacci es una sucesión infinita de números enteros cuyos primeros dos elementos son 0 y 1, los siguientes números son calculados por la suma de los dos anteriores.

## Top-down

A diferencia de bottom-up, aquí se empiezan a hacer los cálculos de n hacia abajo. Además, se aplica una técnica llamada memorización la cual consiste en guardar los resultados previamente calculados, de tal manera que no se tengan que repetir operaciones. Para aplicar la estrategia top-down, se utiliza un diccionario (memoria) el cual va a almacenar valores previamente calculados. Una vez que se realice el cálculo de algún elemento de la sucesión de Fibonacci, éste se va a almacenar ahí.

## Incremental

Es una estrategia que consiste en implementar y probar que sea correcto de manera paulatina, ya que en cada iteración se va agregando información hasta completar la tarea. **Insertion sort**

Insertion sort ordena los elementos manteniendo una sublista de números ordenados empezando por las primeras localidades de la lista. Al principio se

considera que el elemento en la primera posición de la lista está ordenado. Después cada uno de los elementos de la lista se compara con la sublista ordenada para encontrar la posición adecuada.

## Divide y vencerás

Es una estrategia que consiste en: Dividir el problema en subproblemas hasta que son suficientemente simples que se pueden resolver directamente. Después las soluciones son combinadas para generar la solución general del problema.

**Quick sort**

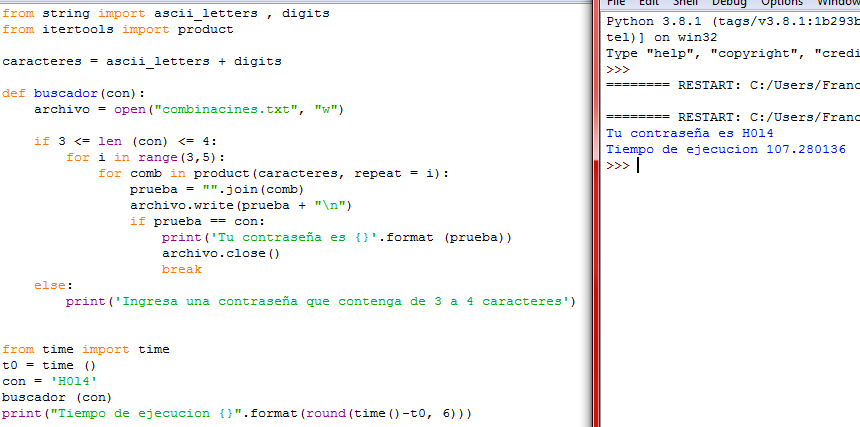
Quicksort es un ejemplo de resolver un problema por medio de la estrategia divide y vencerás. En Quicksort se divide en dos el arreglo que va a ser ordenado y se llama recursivamente para ordenar las divisiones. La parte más importante en Quicksort es la partición de los datos. Lo primero que se necesita es escoger un valor de pivote el cual está encargado de ayudar con la partición de los datos. El objetivo de dividir los datos es mover los que se encuentran en una posición incorrecta con respecto al pivote.

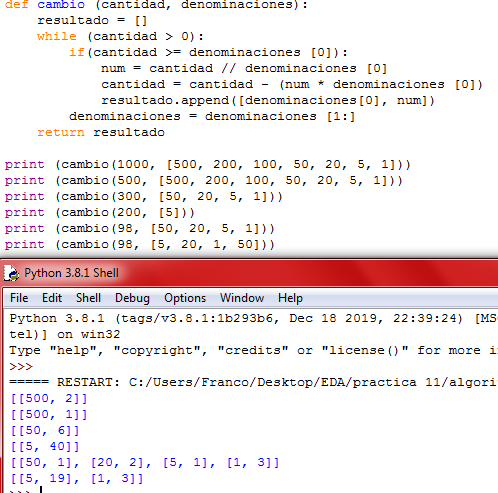
## Modelo RAM

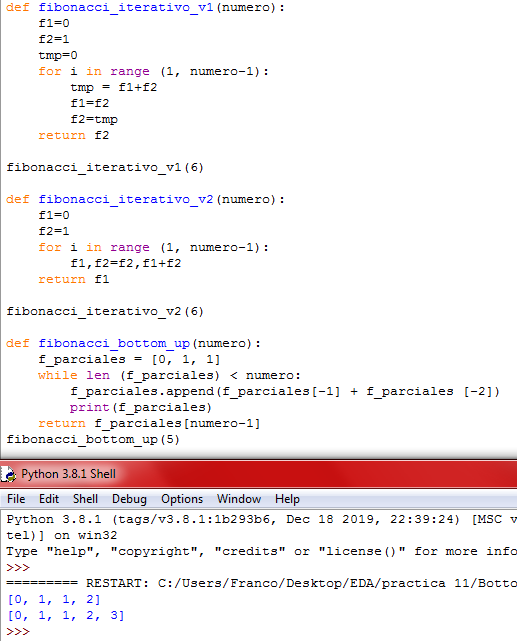
Cuando se realiza un análisis de complejidad utilzando el modelo RAM, se debe contabilizar las veces que se ejecuta una función o un ciclo, en lugar de medir el tiempo de ejecución

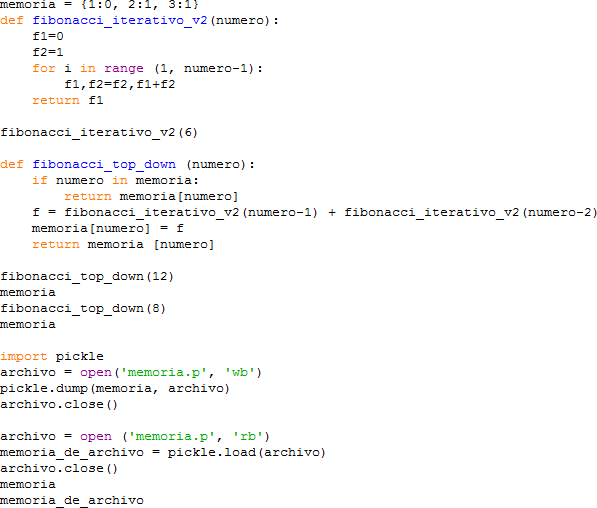
# Desarrollo.

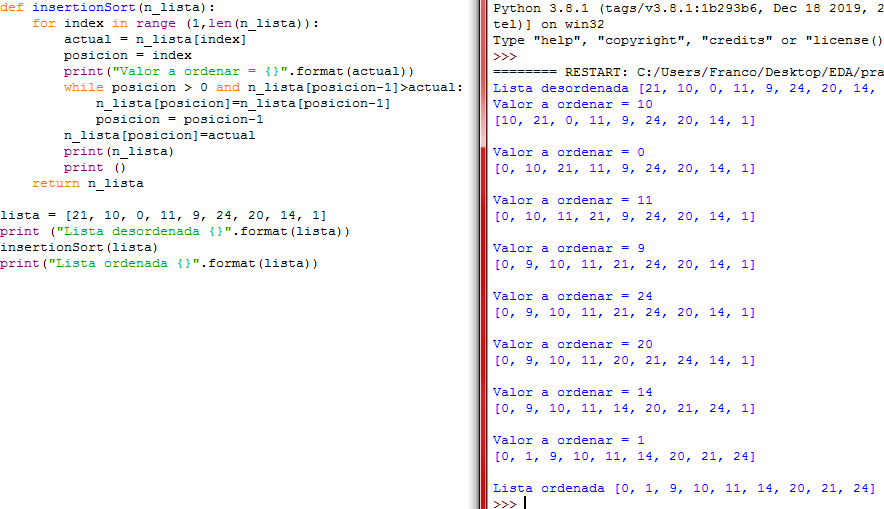
## Fuerza Bruta.

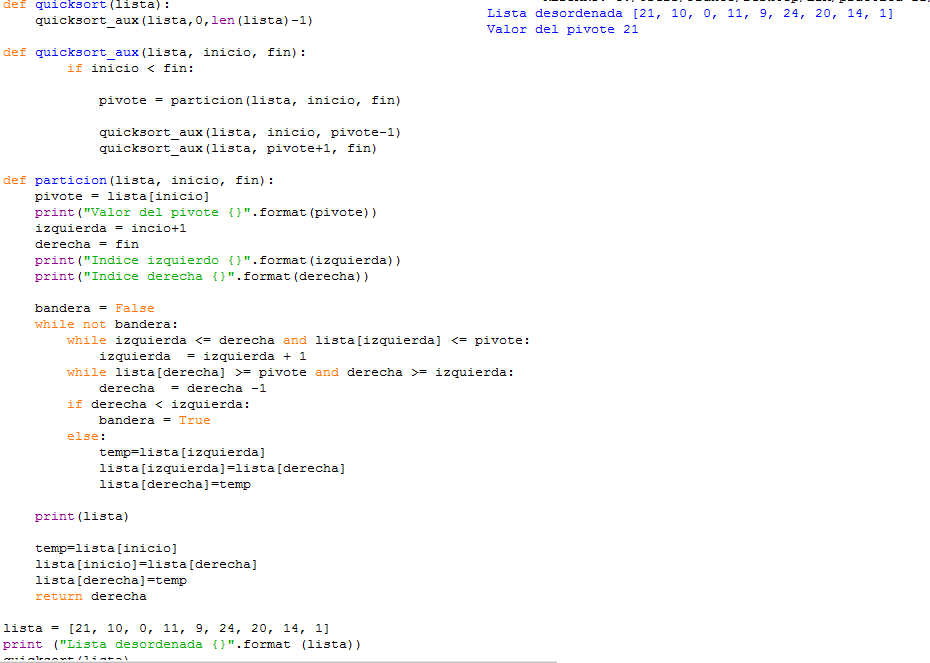


**Algoritmos ávidos (greedy)**

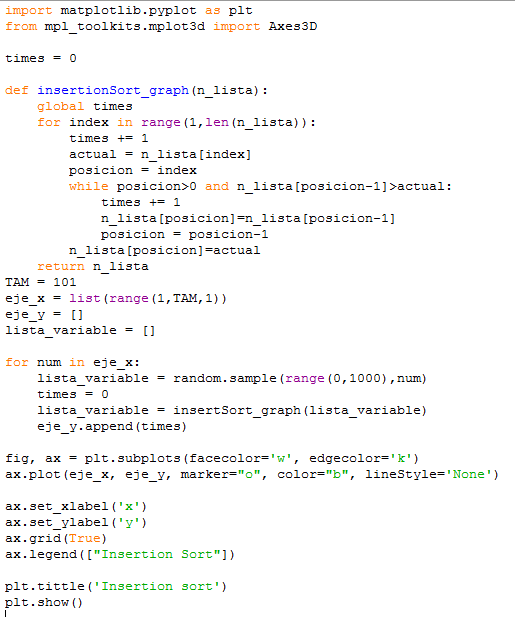
**Bottom-up (programación dinámica)**

**Top-down**

**Incremental**

**Divide y vencerás.**

**Medición y gráficas de los tiempos de ejecución**

**Modelo Ram**

**Conclusiones.**

En esta practica se pudieron apreciar, las diferentes formas de calcular una fuerza utilizando divide y venceras, modelo ram, medición y graficas de los tiempos de ejecucion , entre otros, además, de que se tuvo que buscar un error que contenía este código y así poder arreglarlo de una forma fácil y rápida.

# Referencias.

García Cano, E. E. G. C., & Solano Gálvez, J. A. S. G. (2017, enero 20). Guía práctica de estudio 11:Estrategias para la construcción de algoritmos.

Recuperado 26 de abril de 2020, de [http://lcp02.fi-](http://lcp02.fi-/) b.unam.mx/static/docs/PRACTICAS\_EDA1/eda1\_p11.pdf