|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* |  |
| *Asignatura:* |  |
| *Grupo:* |  |
| *No de Práctica(s):* |  |
| *Integrante(s):* |  |
| *No. de Equipo de cómputo empleado:* |  |
| *No. de Lista o Brigada:* |  |
| *Semestre:* |  |
| *Fecha de entrega:* |  |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

**INTRODUCCIÓN**

Las estrategias para la construcción de algoritmos permiten desarrollar los de manera que los problemas implicados en ellos se resuelvan de manera óptima y eficaz. Entre estas estrategias se encuentran la resolución por fuerza bruta y la construcción de algoritmos ávidos.

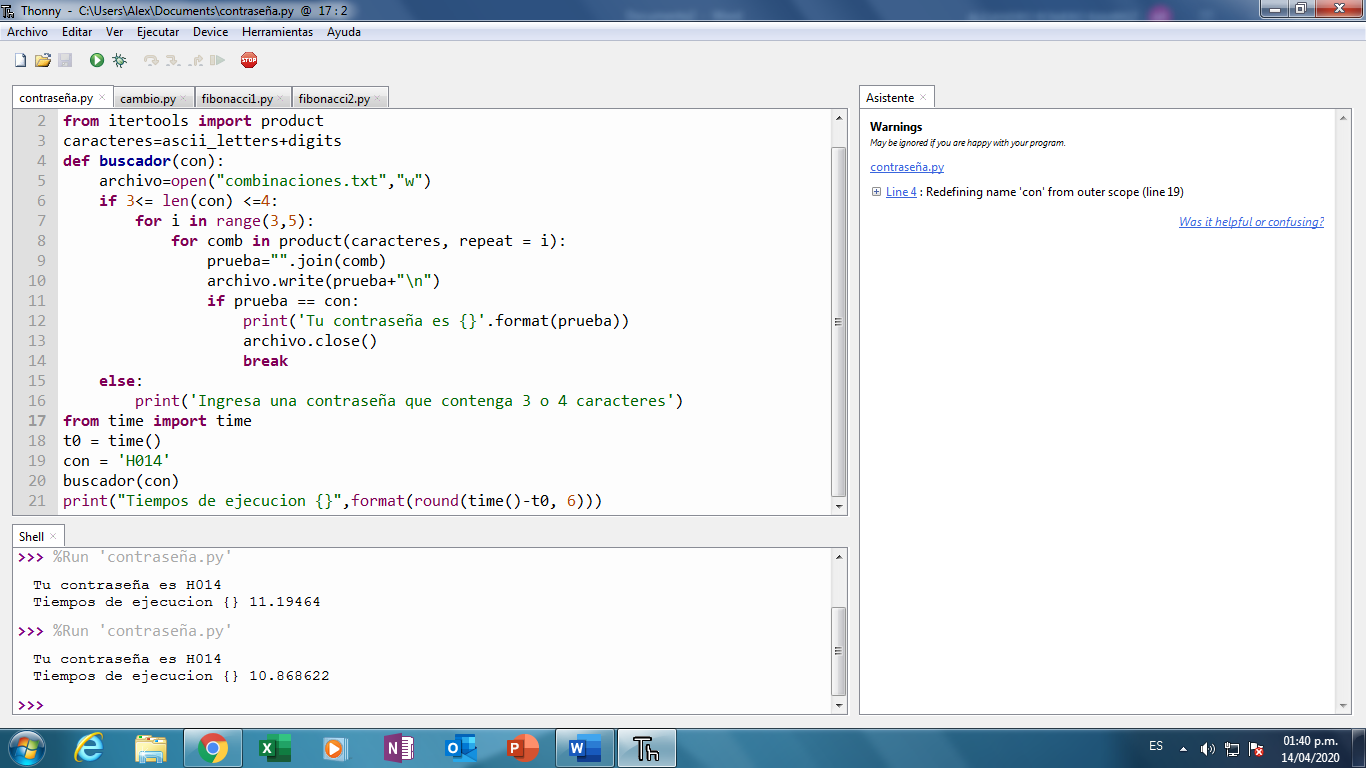
* Fuerza bruta. Se buscan todas las posibilidades que lleven a la resolución del problema, su desventaja es el tiempo que llevan. Por ejemplo, si se quiere encontrar una contraseña con cierto número de caracteres alfanuméricos generando cadenas de cierta longitud, se utiliza esta estrategia.
* Los algoritmos ávidos, a diferencia de la fuerza bruta, va tomando un conjunto de decisiones en específico. La desventaja de éstos es que, una vez tomada una decisión, ya no se vuelve a considerar, y la solución al problema no es la óptima. Por ejemplo, si queremos formar cierta cantidad de dinero con el menor número de billetes y monedas disponibles, primero tendremos que encontrar el billete o moneda más alto que se tenga. Este tipo de algoritmos es muy frecuente en los problemas de optimización.
* Bottom-up. El problema se resuelve a partir de subproblemas resueltos, la solución final se forma a partir de soluciones que, en conjunto, se forman dentro de una tabla.
* Top-down. A diferencia de bottom-up, se hacen los cálculos de n elementos hacia abajo. Se aplica ahí la memorización, que almacena valores previamente calculados.
* Divide y vencerás. Esta estrategia se utiliza para dividir un problema en otros subproblemas de menor tamaño para su óptima solución. Las soluciones de estos problemas combinadas constituyen lo que sería la solución general del problema a resolver.

**OBJETIVO**

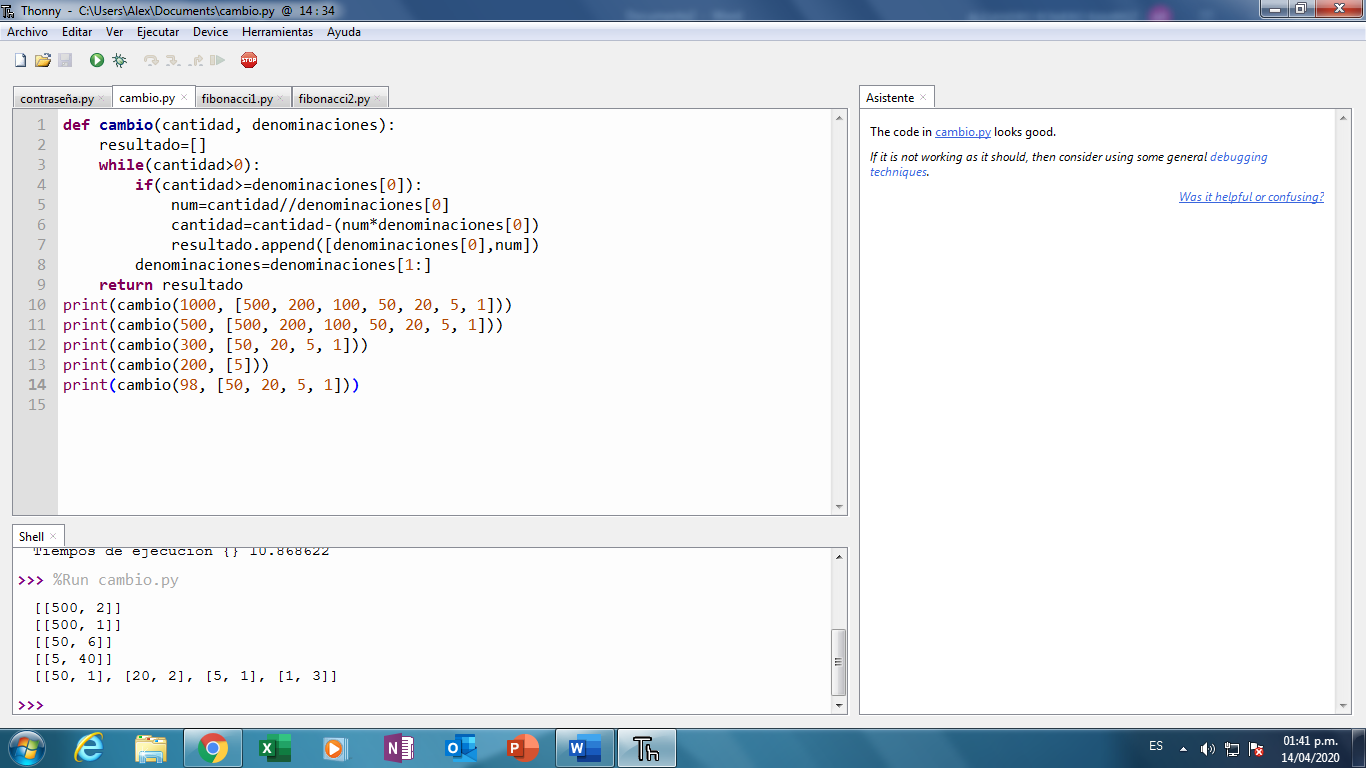
Analizar las características de las estrategias para construir y diseñas algoritmos: fuerza bruta, algoritmos ávidos, bottom-up, top-down y “divide y vencerás”.

**DESARROLLO.**

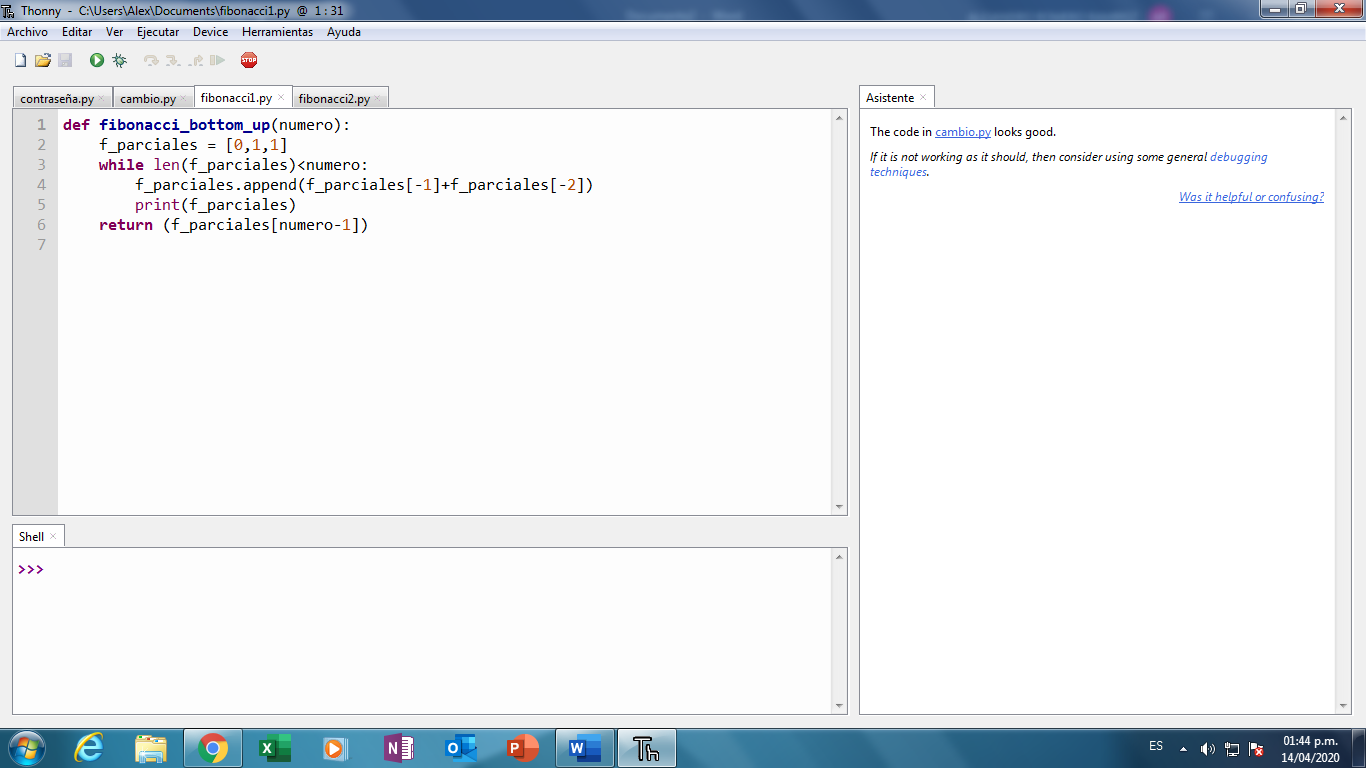
El siguiente código muestra la generación de contraseña por fuerza bruta:



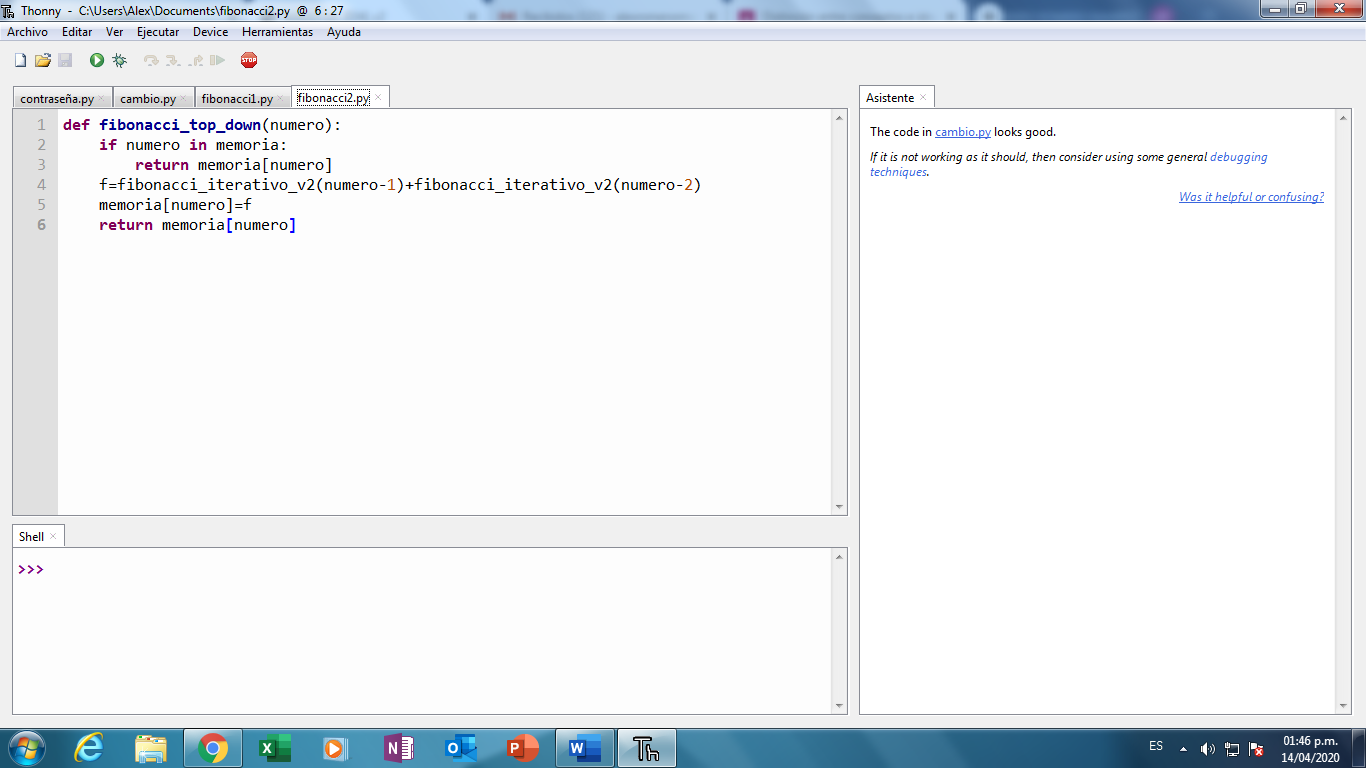
Este código muestra la construcción del cambio en billetes y monedas empleando un algoritmo ávido:



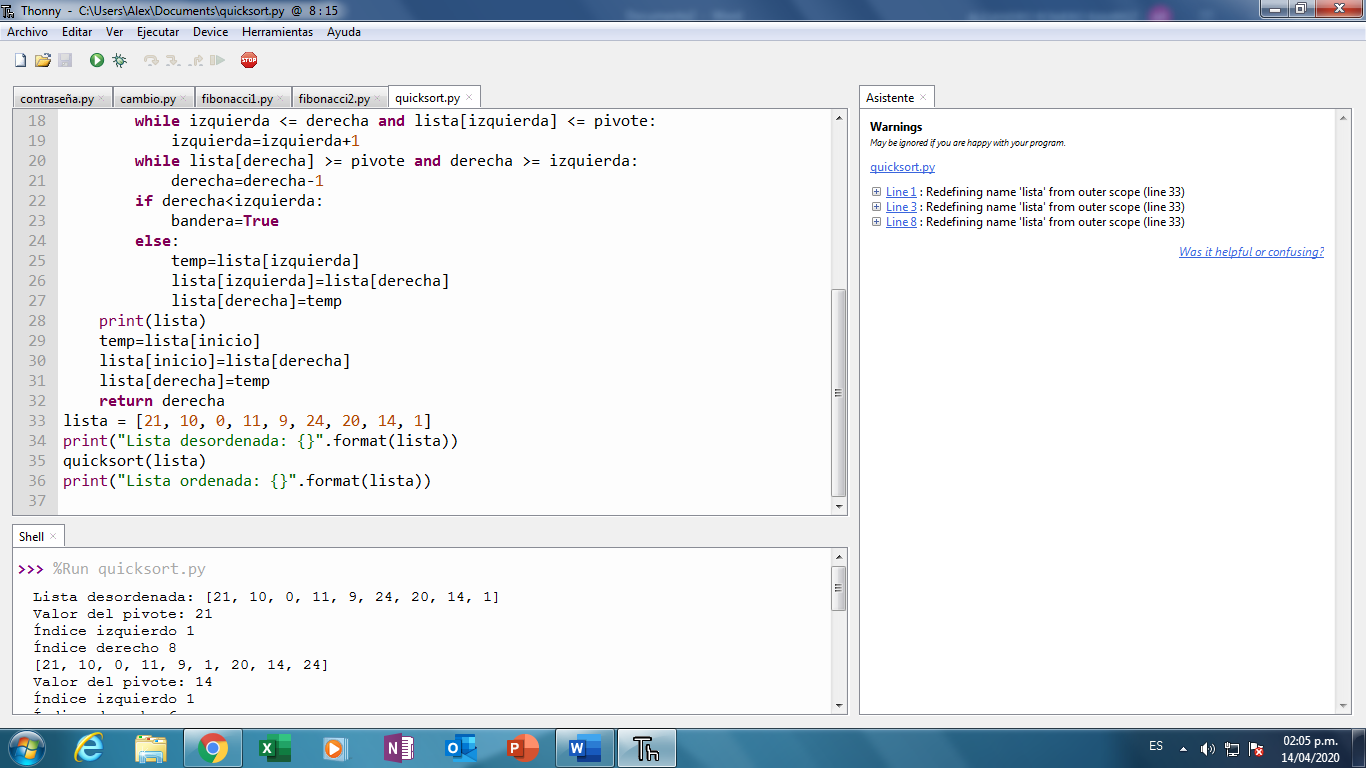
Para aplicar la estrategia bottom-up, se crean soluciones previas, en el caso de la serie de Fibonacci, con el siguiente código:

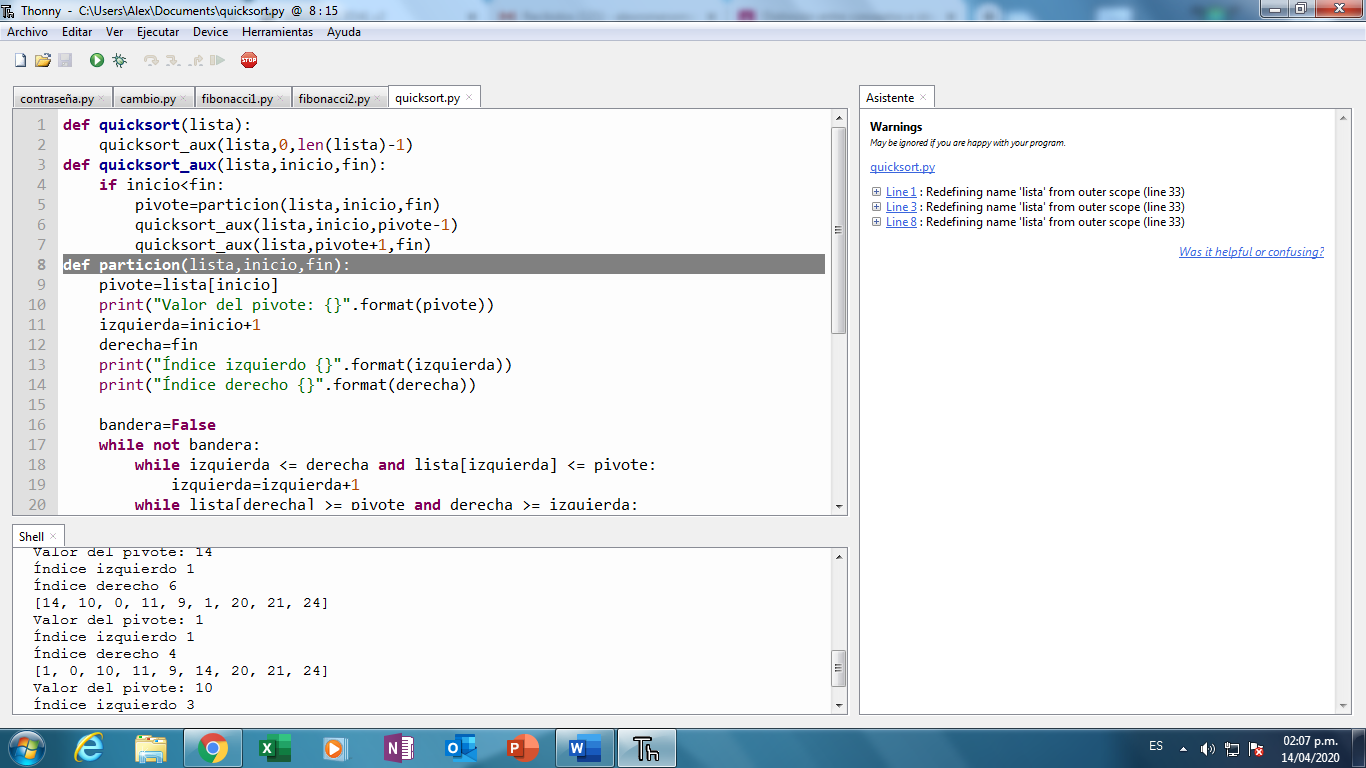


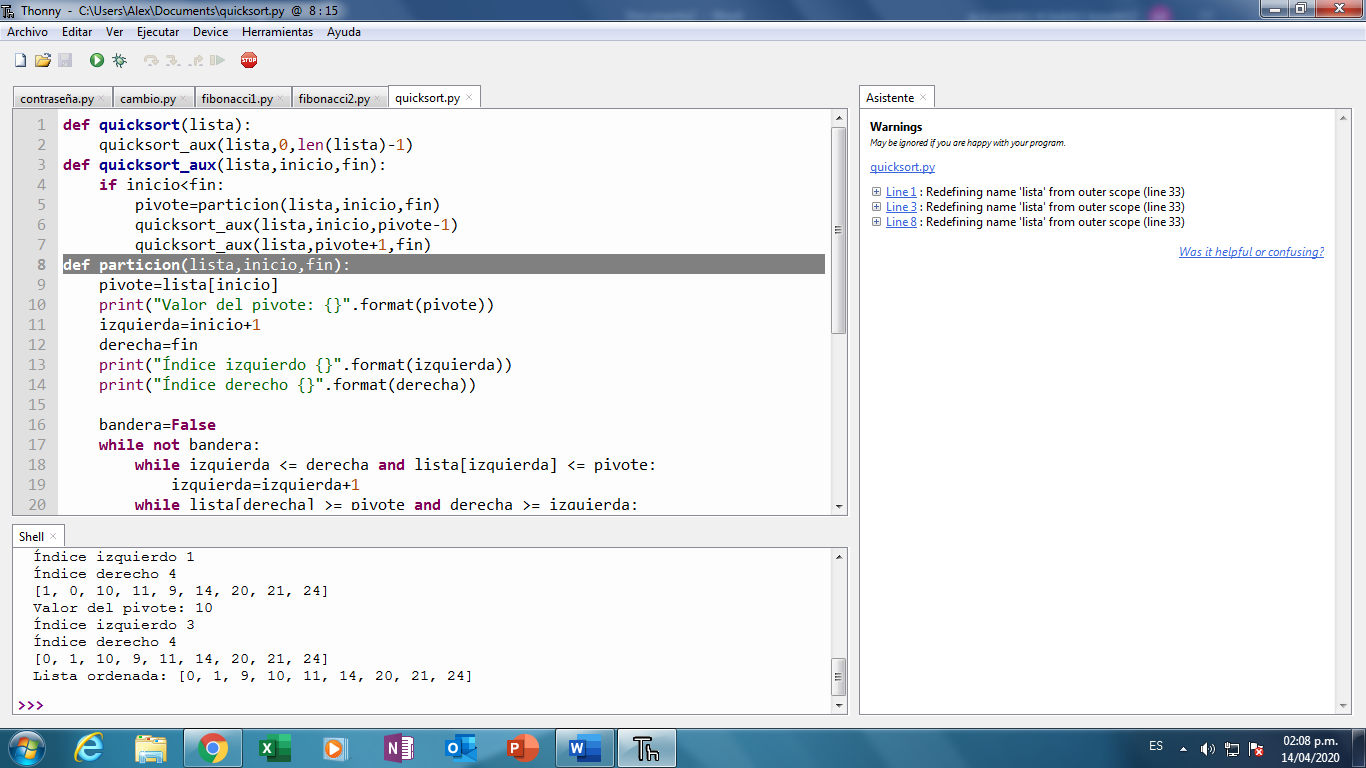
Para aplicar la memorización característica de la estrategia top-down, se emplea el siguiente código:



En este código se ejecuta el ordenamiento de una lista empleando la estrategia de “Divide y vencerás”:







**CONCLUSIONES**

Para su solución óptima, los algoritmos o problemas computacionales se deben subdividir en problemas más pequeños para su análisis en partes, esto es, en partes que el dispositivo informático pueda solucionar gracias a las ya vistas estrategias de construcción.

**BIBLIOGRAFÍA**

Guerequeta, R., & Vallecillo, A. (1998). *Algoritmos ávidos*. Obtenido de Departamento de Lenguajes y Ciencias de la Computación: http://www.lcc.uma.es/~av/Libro/CAP4.pdf

Guerequeta, R., & Vallecillo, A. (1998). *Divide y Vencerás*. Obtenido de Departamento de Lenguajes y Ciencias de la Computación: http://www.lcc.uma.es/~av/Libro/CAP3.pdf