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

Laboratorios de computación

salas A y B



|  |  |
| --- | --- |
| Profesor(a): | M.I. Marco Antonio Martínez Quintana. |
| Asignatura: | Estructura de Datos y Algoritmos I |
| Grupo: | 17 |
| No de Práctica(s): | 12 |
| Integrante(s): | Pineda González Rodrigo |
| No. de Equipo de cómputo empleado: | 26 |
| No. de Lista o Brigada: | 30 |
| Semestre: | 2020-II |
| Fecha de entrega: | 01/05/2020 |
| Observaciones: |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_

# Objetivo:

Aplicar el concepto de recursividad para la solución de problemas.

# Introducción:

El propósito de la recursividad es dividir un problema en problemas más pequeños, de tal manera que la solución del problema se vuelva trivial. Básicamente, la recursión se puede explicar como una función que se llama así misma. Para aplica recursión se deben de cumplir tres reglas:

* Mínimo 1 caso base
* La expansión debe terminar en un caso base.
* La función debe llamarse a sí misma.

## Factorial

La función factorial es una fórmula matemática representada por el signo de exclamación “!”. En la fórmula Factorial se deben multiplicar todos los números enteros y positivos que hay entre el número que aparece en la fórmula y el número 1. La única excepción es el 0, el cual por convención lógica se acordó ser 1.

## Fibonacci

Es una sucesión matemática infinita. Consta de una serie de números naturales que se suman de a 2, a partir de 0 y 1. Básicamente, la sucesión de Fibonacci se realiza sumando siempre los últimos 2 números. Por regla, la sucesión de Fibonacci se escribe así: xn = xn-1 + xn-2.

## Desventajas de la recursividad:

Algunas de las desventajas son:

* Generar una lógica para aplicar la recursión
* Hay una limitación en el número de veces que una función puede ser llamada, tanto en memoria como en tiempo de ejecución.

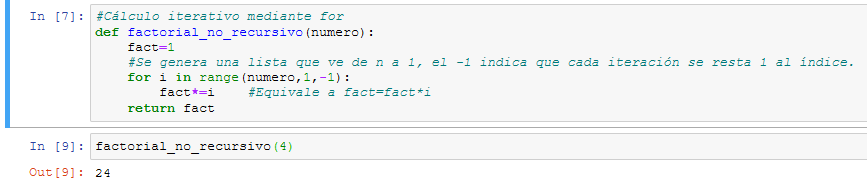
## Repositorio de la práctica:

<https://github.com/RodrigoPG-W/Python12EDAI>

# 

# Desarrollo:

## Factorial



Analizando la forma en que se calcula el factorial en la función pasada se tiene que:

4! =4×3×2×1

Si se remueve el 5 se tiene:

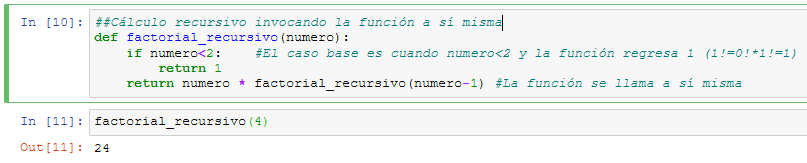
3! =3x2×1=3(3−1)!=3×(2!)

Se puede afirmar que:

3×2! =3×[2(2−1)!]=3×2×(1!)

Si se aplica esto a toda la secuencia, al final tenemos la siguiente expansión:

4! =4(3!) =4×3×(2!) =4×3×2×(1!) =4×3×2×1×(0!) =24

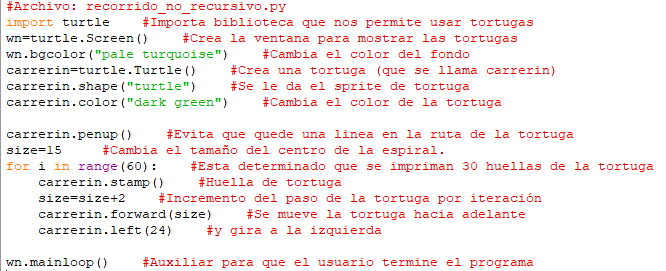


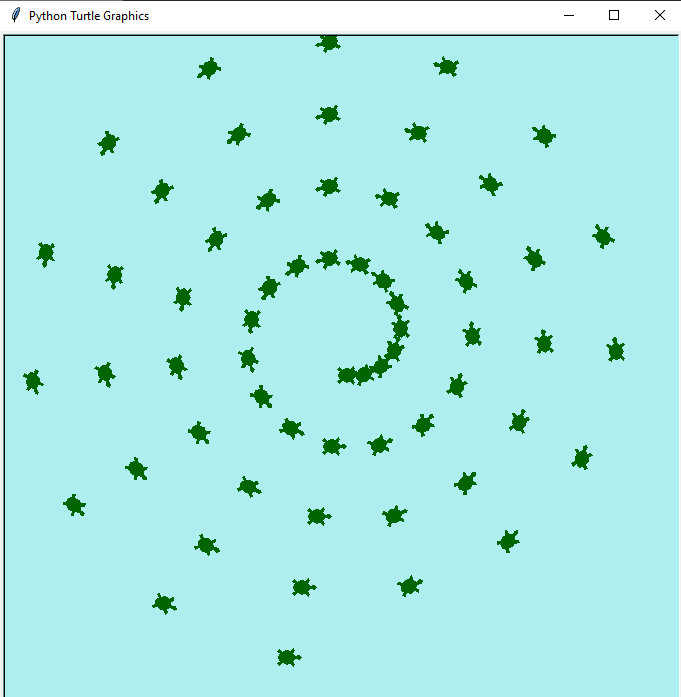
La función se llama a sí misma y toma el lugar del ciclo for usado en la función factorial\_no\_recursivo(). Cada que se llama de nuevo a la función, ésta tiene la copia de las variables locales y el valor de los parámetros.

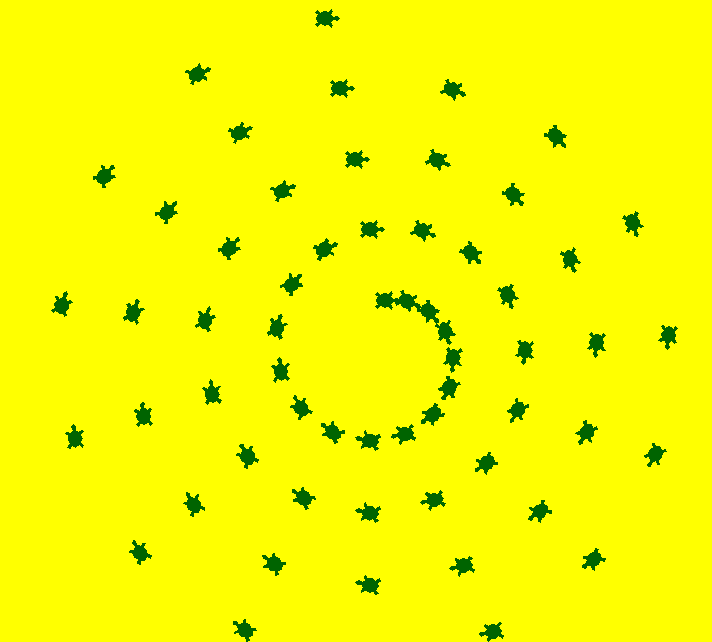
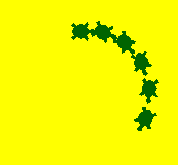
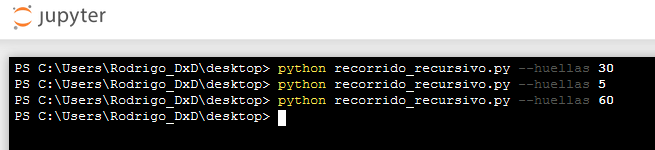
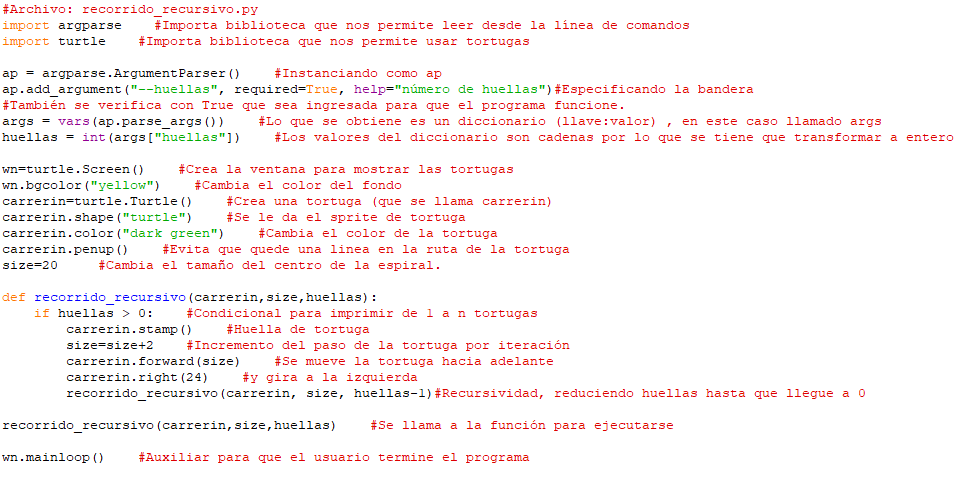
## 

## Huellas de tortuga

El objetivo es hacer que la tortuga deje un determinado número de huellas, cada una de las huellas se va a ir espaciando incrementalmente mientras ésta avanza.







## Fibonacci

## 

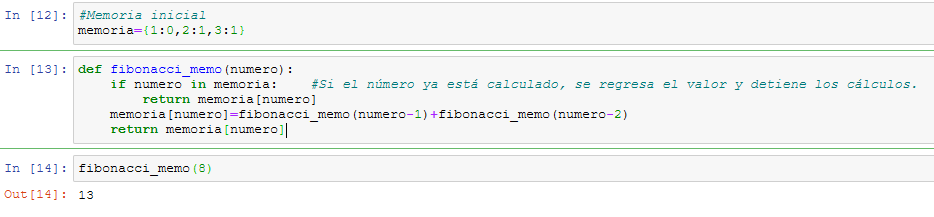
Al igual que en la versión iterativa, se están repitiendo operaciones. Para calcular el elemento 4 se tiene:

f(4)=

(n-1) = f(3) + f(2) + f(1)

(n-2) = f(2) + f(1)

## Memorización



La memoria cambia después de la ejecución. En comparación con la versión iterativa, la función fibonacci\_memo() tiene acceso a la variable memoria, por lo que efectúa menos operaciones. como los resultados se están guardando en la variable memoria, el número de operaciones que se realizan es menor. Para calcular el elemento 5 se tiene:

f(5)=

(n-1) = f(4) + memoria(3) + memoria(2) + memoria(1)

(n-2) = memoria(3)

# Conclusión:

El uso del lenguaje Python se ve bastante favorecido con el apoyo que cuenta, pues se han desarrollado bastantes librerías y el soporte que hay para aprender es muy grande. Un ejemplo fue la librería turtle, que me parece de lo más curiosa y además fue creada precisamente para favorecer el pensamiento como científico computacional. El hecho de poder hacer recursivo un programa tiene muchas ventajas en cuanto la reducción de líneas y optimización de uso de recursos, aunque fuera de problemas diseñados para practicar, me es difícil ver un modelado tan fácil de los problemas para adaptarlos. Además, en el caso de Python, hay un límite en el número de veces que se puede llamar recursivamente una función, si se excede ese límite se genera el error: maximum recursion depth exceeded in comparison. Este límite puede ser modificado, pero no es recomendable. Sin embargo, en el caso de Python, hay un límite en el número de veces que se puede llamar recursivamente una función, si se excede ese límite se genera el error: maximum recursion depth exceeded in comparison. Este límite puede ser modificado, pero no es recomendable. Respecto a la práctica, el uso del símbolo ! para ejecutar los programas resultó innecesario. Desconozco si es porque tengo configurada debidamente la variable de entorno, por el tipo de sistema operativo o porque la práctica está desactualizada.

Referencias:

* *Design and analysis of algorithms*; Prabhakar Gupta y Manish Varshney; PHI Learning, 2012, segunda edición.
* Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest y Clifford Stein; The MIT Press; 2009, tercera edición.
* Problem Solving with Algorithms and Data Structures using Python; Bradley N. Miller y David L. Ranum, Franklin, Beedle & Associates; 2011, segunda edition.
* <http://openbookproject.net/thinkcs/python/english3e/hello_little_turtles.html>
* La función Factorial, (Consultado el 30/04/2020). Recuperado de: <https://factorial.mx/numero-funcion-factorial>
* Fernando Pino, “*¿Qué es la sucesión de Fibonacci?*” (Consultado el 30/04/2020). Recuperado de: <https://www.vix.com/es/btg/curiosidades/4461/que-es-la-sucesion-de-fibonacci>
* Adrian Rosebrock (2018), “*Python, argparse, and command line arguments*” (Consultado el 30/04/2020). Recuperado de: <https://www.pyimagesearch.com/2018/03/12/python-argparse-command-line-arguments/>
* *How to Think Like a Computer Scientist: Learning with Python. Hello, little turtles!* (consultado el 01/05/2020). Recuperado de: <http://openbookproject.net/thinkcs/python/english3e/hello_little_turtles.html>