



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* Marco Antonio Martinez Quintana

*Asignatura:* Estructura de Datos y Algoritmos I

*Grupo:* 17

*No de Práctica(s):* 12

*Integrante(s):* Abrego Abascal Diego

*No. de Equipo de  
cómputo empleado:* -

*No. de Lista o Brigada:* 1

*Semestre:* 2

*Fecha de entrega:* 05/05/2020

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

# Recursividad

## Introducción

La recursividad es implementada en funciones que durante su ejecución se vuelven a llamar a ellas mismas. Esta se utiliza cuando se necesitan generar ciclos que se repiten una y otra vez. Bien implementada la recursividad puede presentar una solución eficaz y elegante a problemas que de otro modo abrían sido más complicados de resolver.

Es importante mencionar que durante la ejecución de una función recursiva el control del programa va pasando hacia cada iteración de la función cada vez mas y mas dentro del ciclo, lo que permite manipular variables muy dentro del ciclo si se establecen las condiciones necesarias. Una vez que se llega a lo más profundo del ciclo por que la condición establecida ya fue satisfecha, el control va retornando de manera inversa, lo que permite realizar más operaciones con las variables obtenidas en lo que el control regresa a la función recursiva superficial, brindando muchas posibilidades a la hora de afrontar retos que requieren de ciclos complejos.

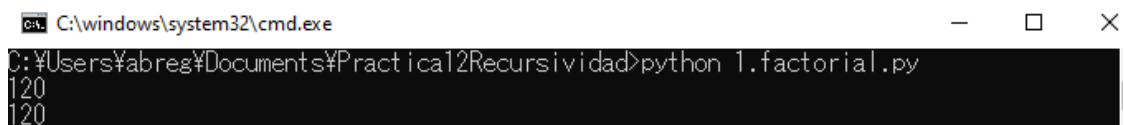
## Objetivo

"El objetivo de esta guía es aplicar el concepto de recursividad para la solución de problemas."

## Desarrollo

### 1. Factorial

```
def factorial_no_recursivo(numero):  
    fact = 1  
    for i in range(numero, 1, -1):  
        fact *= i  
    return fact  
  
print(str(factorial_no_recursivo(5)))  
  
def factorial_recursivo(numero):  
    if numero < 2:  
        return 1  
    return numero * factorial_recursivo(numero - 1)  
  
print(factorial_recursivo(5))  
|
```



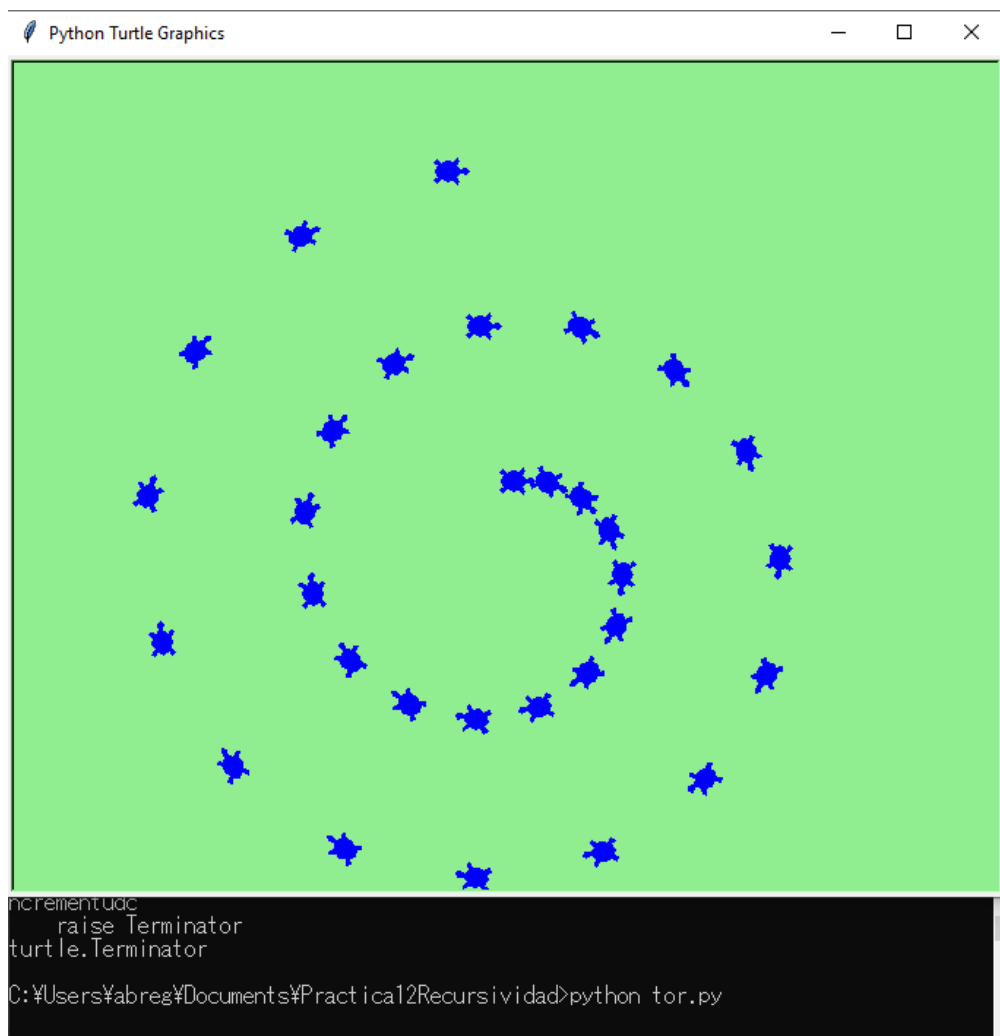
A screenshot of a Windows command prompt window. The title bar shows 'C:\windows\system32\cmd.exe'. The command prompt shows the command 'C:\Users\Yabreg\Documents\Practica12Recursividad>python 1.factorial.py' and the output '120' on two lines.

## 2. Huellas de Tortuga sin Recursividad

```
import turtle
wn = turtle.Screen()
wn.bgcolor("lightgreen")
tess = turtle.Turtle()
tess.shape("turtle")
tess.color("blue")

tess.penup()           # This is new
size = 20
for i in range(30):
    tess.stamp()       # Leave an impression on the canvas
    size = size + 3    # Increase the size on every iteration
    tess.forward(size) # Move tess along
    tess.right(24)     # ... and turn her

wn.mainloop()
```



### 3. Huellas de Tortuga Recursivas

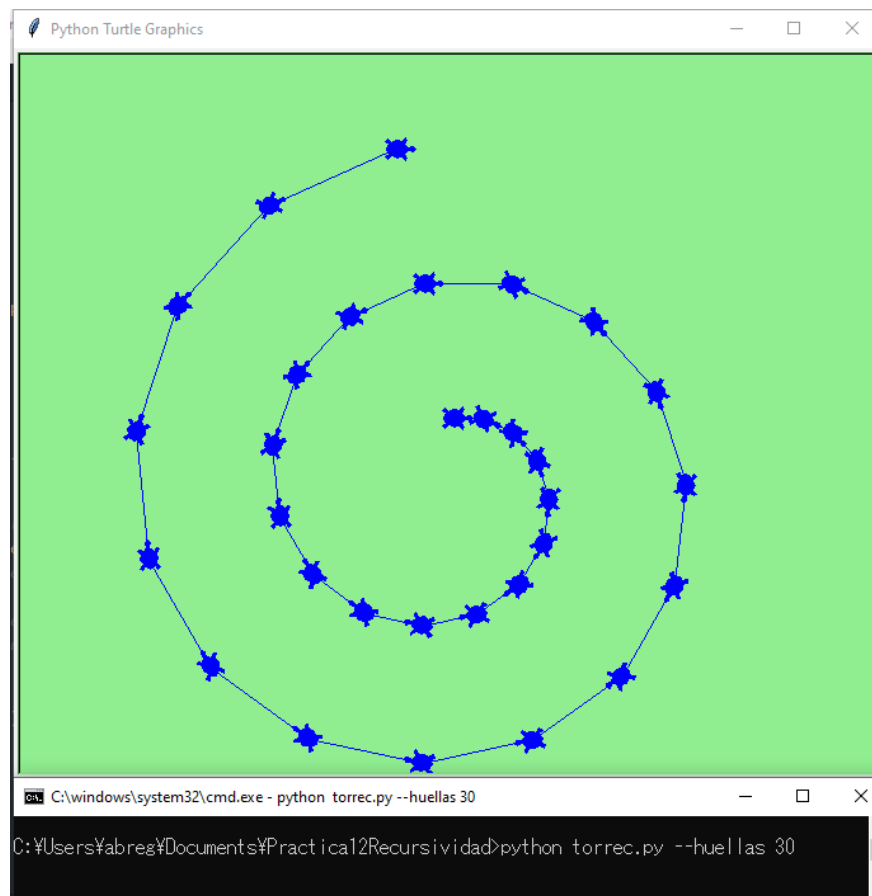
```
import turtle
import argparse
wn = turtle.Screen()
wn.bgcolor("lightgreen")
tess = turtle.Turtle()
tess.shape("turtle")
tess.color("blue")

def recorrido_recursivo(tortuga, espacio, huellas):
    if huellas > 0:
        tortuga.stamp()
        espacio = espacio + 3
        tortuga.forward(espacio)
        tortuga.right(24)
        recorrido_recursivo(tortuga, espacio, huellas-1)

ap = argparse.ArgumentParser()
#El dato de entrada se ingresa con la bandera --huellas
ap.add_argument("--huellas", required=True, help="número de huellas")
#Lo que se obtiene es un diccionario (llave:valor) , en este caso llamado args
args = vars(ap.parse_args())
# Los valores del diccionario son cadenas por lo que se tiene que
# transformar a un entero con la función int()
huellas = int(args["huellas"])

recorrido_recursivo(tess, 20, huellas)

wn.mainloop()
```



#### 4. Huellas de Tortuga Recursivas con parámetros por consola

```
import turtle
import argparse
wn = turtle.Screen()
wn.bgcolor("lightgreen")
tess = turtle.Turtle()
tess.shape("turtle")
tess.color("blue")

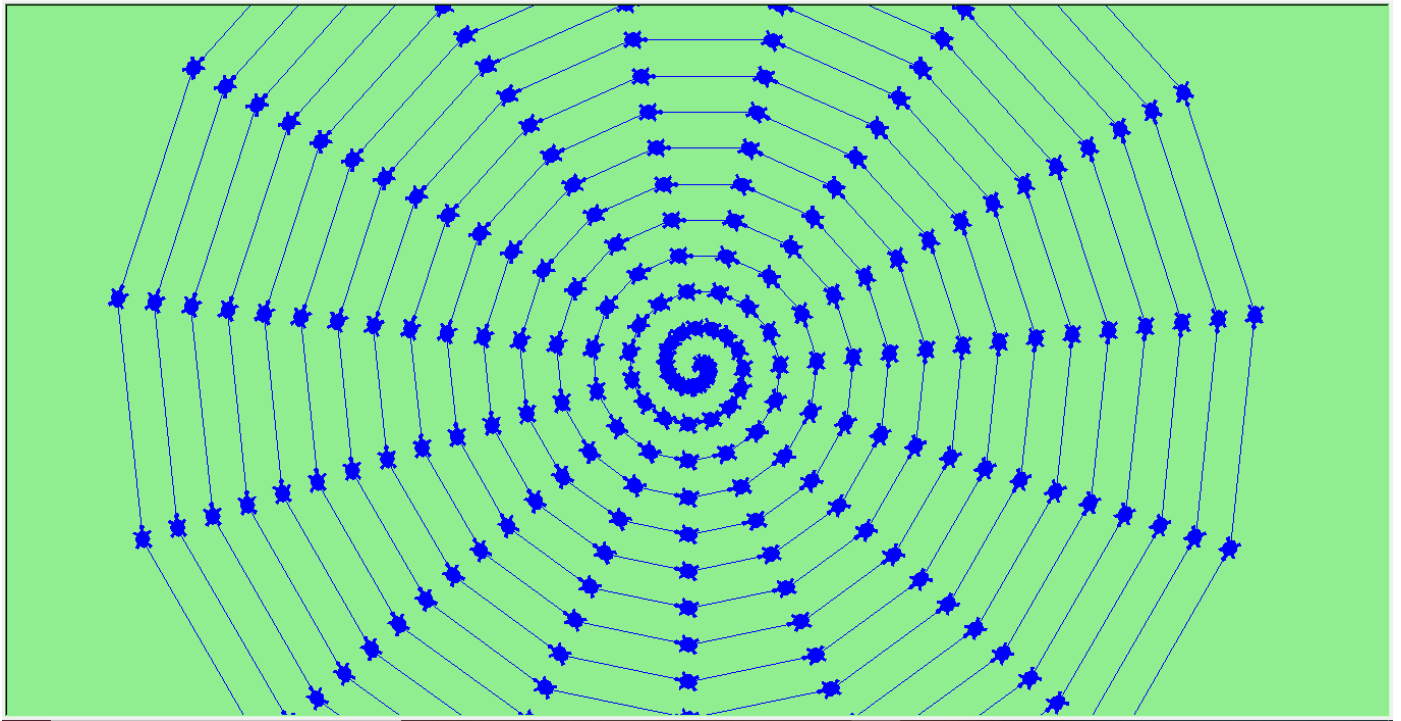
def recorrido_recursivo(tortuga, espacio, huellas):
    if huellas > 0:
        tortuga.stamp()
        espacio = espacio + 1
        tortuga.forward(espacio)
        tortuga.right(24)
        recorrido_recursivo(tortuga, espacio, huellas-1)

ap = argparse.ArgumentParser()
#El dato de entrada se ingresa con la bandera --huellas
ap.add_argument("--huellas", required=True, help="número de huellas")
#Lo que se obtiene es un diccionario (llave:valor) , en este caso llamado args
args = vars(ap.parse_args())
# Los valores del diccionario son cadenas por lo que se tiene que
# transformar a un entero con la función int()
huellas = int(args["huellas"])

recorrido_recursivo(tess,0,huellas)

wn.mainloop()
```

```
C:\windows\system32\cmd.exe
C:\Users\abreg\Documents\Practica12Recursividad>python torrec.py --huellas 100
C:\Users\abreg\Documents\Practica12Recursividad>python torrec.py --huellas 500
Traceback (most recent call last):
  File "torrec.py", line 26, in <module>
    recorrido_recursivo(tess,0,huellas)
  File "torrec.py", line 15, in recorrido_recursivo
    recorrido_recursivo(tortuga, espacio, huellas-1)
  File "torrec.py", line 15, in recorrido_recursivo
    recorrido_recursivo(tortuga, espacio, huellas-1)
  File "torrec.py", line 15, in recorrido_recursivo
    recorrido_recursivo(tortuga, espacio, huellas-1)
  [Previous line repeated 271 more times]
  File "torrec.py", line 13, in recorrido_recursivo
    tortuga.forward(espacio)
  File "C:\Users\abreg\Documents\Practica12Recursividad\turtle.py", line 1519, in forward
    self._go(distance)
  File "C:\Users\abreg\Documents\Practica12Recursividad\turtle.py", line 1491, in _go
    self._goto(ende)
  File "C:\Users\abreg\Documents\Practica12Recursividad\turtle.py", line 2911, in _goto
    screen._drawline(self.drawingLineItem,
  File "C:\Users\abreg\Documents\Practica12Recursividad\turtle.py", line 522, in _drawline
    self.cv.coords(lineitem, *cl)
  File "<string>", line 1, in coords
  File "C:\Program Files\Python38\lib\tkinter\__init__.py", line 2761, in coords
    self.tk.call((self._w, 'coords') + args))
_tkinter.TclError: invalid command name ".!canvas"
```



## 5. Sucesión de Fibonacci de manera recursiva y no recursiva

```
def fibonacci_iterativo_v2(numero):
    f1=0
    f2=1
    for i in range(1, numero-1):
        f1,f2=f2,f1+f2 #Asignación paralela
    return f2

print(str(fibonacci_iterativo_v2(13)))

def fibonacci_recursivo(numero):
    if numero ==1:
        return 0
    if numero == 2 or numero == 3:
        return 1

    return fibonacci_recursivo(numero-1) + fibonacci_recursivo(numero-2)

print(str(fibonacci_recursivo(13)))

memoria = {1:0,2:1,3:1}

def fibonacci_memo(numero):
    if numero in memoria:
        return memoria[numero]
    memoria[numero] = fibonacci_memo(numero-1) + fibonacci_memo(numero-2)
    print(memoria)
    return memoria[numero]

print(fibonacci_memo(13))
print(fibonacci_memo(11))
```

```
C:\Users\abreg\Documents\Practica12Recursividad>python 4.fibonacci.py
144
144
[1: 0, 2: 1, 3: 1, 4: 2]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8, 8: 13]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8, 8: 13, 9: 21]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8, 8: 13, 9: 21, 10: 34]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8, 8: 13, 9: 21, 10: 34, 11: 55]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8, 8: 13, 9: 21, 10: 34, 11: 55, 12: 89]
[1: 0, 2: 1, 3: 1, 4: 2, 5: 3, 6: 5, 7: 8, 8: 13, 9: 21, 10: 34, 11: 55, 12: 89, 13: 144]
144
55
```

## Conclusión

Con la elaboración de esta práctica se revisó el concepto y la aplicación de la recursividad, la cual permite hacer que una función se vuelva a llamar a si misma generando ciclos que si se saben utilizar pueden añadir muchas mas alternativas a la hora de solucionar problemas.

La recursividad fue usada para calcular la sucesión de Fibonacci, así como el factorial de un determinado número. También se revisó la biblioteca "*turtle*" en la cual se creaban objetos(tortugas) los cuales se podían manipular mediante las funciones específicas de la librería. Se uso un ciclo for y una función recursiva para dirigir el comportamiento de la tortuga permitiendo apreciar una vez más un uso de la recursividad, pero en esta ocasión llamando a mas funciones cada vez.

Por lo mencionado anteriormente y por la correcta realización de la práctica, concluyo que los objetivos se cumplieron correctamente.

## Bibliografía

- [http://lcp02.fi-b.unam.mx/static/docs/PRACTICAS\\_EDA1/eda1\\_p12.pdf](http://lcp02.fi-b.unam.mx/static/docs/PRACTICAS_EDA1/eda1_p12.pdf)
- <https://www.python.org>
- [http://openbookproject.net/thinkcs/python/english3e/hello\\_little\\_turtles.html](http://openbookproject.net/thinkcs/python/english3e/hello_little_turtles.html)