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

# Laboratorios de computación salas A y B

*Profesor: Martínez Quintana Marco Antonio*

*Asignatura: Estructuras de Datos y Algoritmos I*

*Grupo: 17*

*No de Práctica(s): 12*

*Integrante(s): De León Arias Emiliano*

*No. de Equipo de cómputo empleado: 37*

*No. de Lista o Brigada: 13*

*Semestre: 2020-2*

*Fecha de entrega: 3 mayo 2020*

*Observaciones:* CALIFICACIÓN: _____

# Objetivo

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

# Introducción

La recursividad se basa en subdividir un problema en otros más pequeños para así llegar a su solución de una manera más simple.

Para poder aplicar recursión se deben aplicar tres casos:

-Debe de haber uno o más casos base.
-La expansión debe terminar en un caso base.
-La función se debe llamar a sí misma.

Se debe tener cuidado en Python ya que la recursividad no es ilimitada, tiene un límite, si bien se puede modificar ese límite, no es muy recomendable hacerlo.

Es por ello por lo que también se deben tomar en cuenta las desventajas que tiene la recursividad, las cuales son principalmente dos:

-A veces es complejo generar la lógica para aplicar recursión.
-Hay una limitación en el número de veces que una función puede ser llamada, tanto en memoria como en tiempo de ejecución.

# Desarrollo

a) Código

```
def factorial_no_recursivo(numero):
    fact=1
    for i in range(numero,1,-1):
        fact *=i
    return fact
factorial_no_recursivo(5)

def factorial_recursivo(numero):
    if numero<2:
        return 1
    return numero*factorial_recursivo(numero -1)
factorial_recursivo(5)

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)
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)
    return memoria[numero]
fibonacci_memo(13)

#Archivo:recorrido_no_recursivo.py

for i in range(30):
    tess.stamp()
    size=size+3
    tess.forward(size)
    tess.right(24)
```

```
#Archivo:recorrido_recursivo.py

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)
```

## Conclusiones

La recursividad es una herramienta muy útil que nos permitirá resolver los problemas de una manera eficaz y sencilla, solo se debe cuidar cumplir con los requisitos para poder utilizarla y analizar antes de codificar si en verdad es necesario utilizarla o lo mas conveniente para llegar a la solución deseada.

## Bibliografía

Laboratorios A y B, Practica 12 Recursividad, consultado el 1 mayo de 2020, de file: http://lcp02.fi-b.unam.mx/