**Práctica 2**

*Algoritmos básicos II*

**Unidad académica:** Análisis de Algoritmos

**Profesor a cargo:** Dra. Sandra Díaz Santiago

**Grupo:** 3CV1

**Realizada por:**

* Medina Juárez Jesús Booz
* Ríos Altamirano Alam Yael

**Sesión de laboratorio:** 28 de febrero del 2018  
**Fecha de entrega:** 2 de marzo del 2018

# Diseño de algoritmos

## Ejercicio 1

maximominimo(A[n]) Costo #Operaciones

res [2] := A[0] // arreglo de dos posiciones, ambas inician en A[0] C1 1

for i := 0 to n C2 n

if(A[n] < res[min]) C3 n

res[min] := A[n] C4 n

if(A[n] > res[max]) C3 n

res[max] := A[n] C4 n

return res C5 1

Tiempo polinomial y notación asintótica

## Ejercicio 2

Exponenciación modular rápida (logarítmica)

exponenciacionModularRapida(a, m, n) Costo #Operaciones

res := 1 C1 1

a := a mod n C2 1

while (m>0) C3

if (odd(m)) C4

res = (res \* a) mod n C5

m = m >> 1 C6

a = (a \* a) mod n C7

return res C8 1

Tiempo polinomial y notación asintótica

## Ejercicio 3

selectionsort(A[]) Costo #Operaciones

for i := 0 to n C1 n

min := I C2 n

for j := (i + 1) to n C3

if A[j] < A[i] C4

min := j C5

A[i],A[min] := A[min],A[i] C6 n

return A C7 1

Tiempo polinomial y notación asintótica

## Ejercicio 4

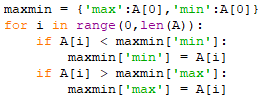
Ejemplo de aplicación de algoritmo merge sort para el ordenamiento de alturas de ramas de árboles

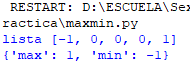
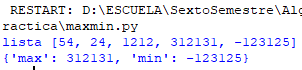
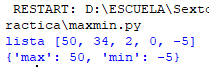


# Ejercicios de programación

## Ejercicio 1

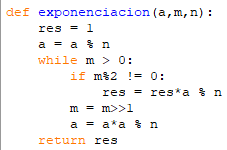
Selección de máximo y mínimo implementada en Python 2.7

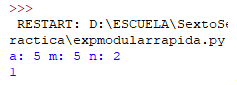
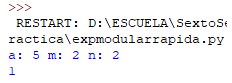
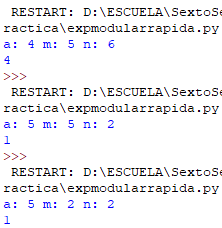




## Ejercicio 2

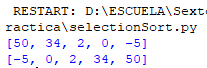
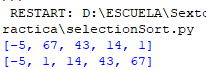
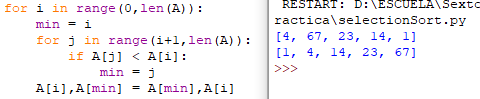
Exponenciación modular rápida implementada en Python 2.7





## Ejercicio 3

Selection sort implementado en Python 2.7



## Ejercicio 4

Merge sort implementado en Python 2.7

