Dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

ESCUELA SUPERIOR DE CÓMPUTO

INSTITUTO POLITÉCNICO NACIONAL



**Realizado por:** Pérez Marcelo Cristopher

**Asignatura:** Análisis y Diseño de Algoritmos

**Grupo:** 3CV4

**Fecha:** 11-06-2023

Ejercicio de laboratorio no.4

# Ejercicio de laboratorio no.4

Instrucciones: A partir de los códigos proporcionados en el archivo adjunto, realiza las siguientes actividades.

1. Para el código que resuelve el **laberinto en C**, propón al menos 5 modificaciones en donde dicho laberinto sea más grande, más difícil o de plano imposible de resolver. Comprueba con el programa qué ocurre en todos los casos y reporta si hubo cambios abruptos en el tiempo de ejecución.
2. Revisa el programa equivalente en Python ¿Qué diferencias observas? Realiza los mismos cambios de laberintos que en C y reporta si hay cambios abruptos en el tiempo de ejecución.

Primer cambio: Incremento de las filas.

Segundo cambio: Incremento en las columnas.

Tercer cambio: Diferente inicio.

Cuarto cambio: Diferente fin.

Quinto cambio: Mayor cantidad de callejones.

A continuación, se muestran los resultados.

Captura de pantalla de computadora

Descripción generada automáticamenteCalendario

Descripción generada automáticamente

1. Ejecuta el programa **find\_min\_max.c**, describe las partes en las que se hace el método de Divide and Conquer.
2. Ejecuta su programa equivalente en python ¿Qué diferencias observas?

Tanto el lenguaje implementado en lenguaje C como en Python trabajan de la misma manera. Como característica del método Divide and Conquer tenemos que trabajar con recursividad y especificar los casos bases, estos serían cuando únicamente trabajamos con un número y cuando el arreglo solo tiene dos elementos, en cuyo caso solo realizaríamos una comparación para saber cuál es el número menor y mayor respectivo a sus funciones. Estos serían el punto de corto para romper con la recursividad.

Las diferencias entre los distintos lenguajes no son notables ya que utilizan la misma lógica (sin mencionar las evidentes diferencias de su sintaxis).

Sin embargo, algo que puedo mencionar es que para el caso de Python cuando ingresaba un arreglo de 1000 elementos el programa colapsaba debido a que excedía el número de comparaciones permitidas.

Imagen de la pantalla de un celular

Descripción generada automáticamente con confianza media

Y en el caso del lenguaje C no se presentaba este problema. Por ello, la comparación de los tiempos de ejecución se realizó con solo 100 elementos en ambos casos.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

1. Texto

   Descripción generada automáticamenteTexto

   Descripción generada automáticamenteAumenta el tamaño del arreglo, genera un arreglo de hasta 10 mil elementos. Compara los tiempos de ejecución en C y en Python.

El tiempo de ejecución en Python es mayor que en el lenguaje C (0.00014 > 0.000035)

1. Ejecuta el programa **quicksort.c**, describe las partes en las que se hace el método de Divide and Conquer.
2. Describe la lógica del algoritmo, puedes utilizar cualquier fuente de información.

El método Quick Sort es actualmente el mas eficiente y veloz de los método de ordenación interna. Quicksort es un algoritmo basado en la técnica de divide y vencerás, que permite, en promedio, ordenar n elementos en un tiempo proporcional a n log n.

Para ésto utiliza la **recursividad**.

* + - Dada una lista, elegir uno de sus elementos, que llamamos **pivot**
    - Dividir la lista en dos sublistas:
      * una con los elementos "menores"
      * otra con los elementos "mayores"
    - Ordenar recursívamente ambas sublistas
    - Armar la lista resultado como: menoresOrdenados + pivot + mayoresOrdenados
    - El algoritmo está dividido en 3 funciones.

Recuperado de: <https://sites.google.com/site/programacioniiuno/temario/unidad-6---anlisis-de-algoritmos/algoritmo-quicksort>

1. Ejecuta su programa equivalente en python ¿Qué diferencias observas?

Las diferencias que tienen los dos programas, además de la diferencia sintáctica, es que el código de C tiene un módulo especial para hacer el intercambio de los elementos *void swap(int\* a, int\* b)* en dónde se utiliza una variable auxiliar para hacer el cambio. Por otro lado, Python tiene una manera más sencilla de hacer dicho intercambio: *void swap(int\* a, int\* b)*

1. Aumenta el tamaño del arreglo, genera un arreglo de hasta 10 mil elementos. Compara los tiempos de ejecución en C y en Python.

Captura de pantalla de computadora

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

El tiempo de ejecución de Python es mayor en comparación al de C (0.345>0.004).

1. Ejecuta el programa **mergesort.c**, describe las partes en las que se hace el método de Divide and Conquer.
2. Describe la lógica de programación del algoritmo, puedes usar cualquier fuente de información.
   * + El paso de dividir tarda un tiempo constante, sin importar el tamaño del subarreglo. Después de todo, el paso de dividir simplemente calcula el punto medio q de los índices p y r. Recuerda que en la notación Θ grande, indicamos un tiempo constante por O(1)
     + El paso de vencer, en donde ordenamos dos subarreglos de aproximadamente n/2 elementos cada uno, se tarda una cierta cantidad de tiempo, pero vamos a contabilizar ese tiempo cuando consideremos los subproblemas.
     + El paso de combinar mezcla un total de n elementos y se tarda un tiempo O(n)

Combinando los tiempos recursivos de los subproblemas y los tiempos de combinaciones finales obtenemos que *mergeSort* se ejecuta en tiempo de **O(n\*log2n).**

Este algoritmo del tipo "Divide y Vencerás" se rige bajo la siguiente lógica:

**Caso Base:** una lista vacía o de un elemento ya está ordenada

**Caso recursivo:** consiste en partir una estructura en mitades, ordenar cada mitad y luego intercalar ordenadamente ambas mitades.

Cada mitad se ordena aplicando el mismo método.

Esto es:

1. Dividir en “mitades”
2. Ordenar la primera mitad
3. Ordenar la segunda mitad
4. Intercalar las mitades ordenadas

Recuperado de:

<https://es.khanacademy.org/computing/computer-science/algorithms/merge-sort/a/analysis-of-merge-sort>

<http://www.cs.uns.edu.ar/~mlg/ipoo/downloads/Presentaciones%20y%20Apuntes/0.Java/0.10%20Merge%20Sort%20&%20Quick%20Sort%20&%20Matriz%20Racional.pdf>

1. Ejecuta su programa equivalente en python ¿Qué diferencias observas?

En el código de Python después de dividir el arreglo inicial se repite recursivamente el proceso primero para el lado izquierdo y luego el derecho

para después copiarlos en los arreglos auxiliares, mientras que en el de C se hacen los arreglos auxiliares primero y después se ordenan.

1. Captura de pantalla de computadora

   Descripción generada automáticamenteAumenta el tamaño del arreglo, genera un arreglo de hasta 10 mil elementos. Compara los tiempos de ejecución en C y en Python.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

El tiempo de ejecución de Python es mayor al de lenguaje C (0.511>0.005 segundos)