Unidad 1

Análisis de Complejidad Computacional

Docente:

M. en C. Erika Sánchez-Femat

Practica 1

por

Giovanna Inosuli Campos Flores

September 16, 2023

1 Objetivo de la Práctica

El objetivo principal de esta práctica es que los estudiantes adquieran una comprensión sólida y profunda de los conceptos clave relacionados con la complejidad computacional de los algoritmos. Los tres casos principales a analizar, mejor caso, peor caso y caso promedio, permiten una evaluación completa del rendimiento de un algoritmo bajo diferentes circunstancias.

1.1 Objetivos Particulares

- **Comprender la Variabilidad de la Ejecución:** Los algoritmos pueden comportarse de manera diferente según las características de los datos de entrada. Al analizar el mejor, peor y caso promedio, los estudiantes aprenderán a apreciar cómo un algoritmo puede variar en su rendimiento.
- **Aplicar Conceptos de Complejidad Computacional:** La práctica ayudará a los estudiantes aplicar conceptos fundamentales de la teoría de la complejidad computacional, como la notación Ogrande (Big O), para describir y comparar el rendimiento de los algoritmos.
- **Desarrollar Habilidades de Análisis Crítico:** A través del análisis de diferentes casos, los estudiantes mejorarán sus habilidades de análisis crítico, aprendiendo a identificar situaciones en las que un algoritmo puede ser más eficiente o ineficiente.
- **Preparación para Desarrollo de Algoritmos Eficientes:** Al comprender los casos de mejor, peor y caso promedio, los estudiantes estarán mejor preparados para diseñar y seleccionar algoritmos eficientes en situaciones reales, lo que es fundamental en la resolución de problemas computacionales.
- **Promover la Resolución de Problemas:** A través de la práctica, los estudiantes aprenderán a abordar y resolver problemas computacionales de manera más efectiva, seleccionando o adaptando algoritmos en función de las restricciones de tiempo y recursos.

2 Desarrollo de la Práctica

2.1 Implementación de los algoritmos

se utilizo pyton para escribir los metodos de ordamientos

- Burbuja El ordenamiento de burbuja es un algoritmo de ordenamiento simple y popular que funciona iterativamente comparando pares de elementos adyacentes en una lista y realizando intercambios si están en el orden incorrecto.Este proceso se repite hasta que la lista esté ordenada por completo. El algoritmo burbuja se compone de 2 bucles, uno dentro del

otro.En Python es posible realizar la asignación simultánea. La instrucción a,b=b,a dará lugar a que se realicen dos instrucciones de asignación al mismo tiempo (véase la Figura). Usando la asignación simultánea, la operación de intercambio se puede hacer en una sola instrucción

```
def ord_burbuja(arreglo):
    n = len(arreglo)
    for i in range(n-1):
        print(f"{i+1} Recorrido (i = {i})")
        for j in range(n-1-i):
            print("j =", j)
            if arreglo[j] > arreglo[j+1]:
                arreglo[j], arreglo[j+1] = arreglo[j+1], arreglo[j]
```

- Burbujas Optimizada

  El ordenamiento de burbuja optimizado es una variante del algoritmo de ordenamiento de burbuja que implementa algunas estrategias para mejorar su eficiencia. Aunque el ordenamiento de burbuja no es el algoritmo más eficiente en términos de tiempo, estas optimizaciones pueden hacer que funcione mejor en ciertos casos.

```
def burbuja_optimus(arreglo):
    n = len(arreglo)

    for i in range(n-1):
        intercambio = False

        for j in range(n-1-i):
            if arreglo[j] > arreglo[j+1] :
                arreglo[j], arreglo[j+1] = arreglo[j+1], arreglo[j]
                intercambio = True

        if intercambio == False:
            break
```

2.1.1 Análisis de Casos

Calculamos el mejor, peor y caso promedio para ambos algoritmos de ordenamiento, dando 3 ejemplos de datos de entrada para cada caso. los ejemplos que utilizamos fueron los siguentes hay que aclara que nuestro programa el usario pone el tamaño de arreglo y tambien pone los elementos

de la lista para realizar el ordenamientos

```
Que tamaño sera la lista:
10
Ingrese el elemento 1
1
Ingrese el elemento 2
2
Ingrese el elemento 3
3
Ingrese el elemento 4
4
Ingrese el elemento 5
5
Ingrese el elemento 6
6
Ingrese el elemento 7
7
Ingrese el elemento 8
8
Ingrese el elemento 9
9
Ingrese el elemento 10
10
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
```

- Mejor caso: O(n)

1. Burbuja

```
Burbuja

Lista
[1.0, 2.0, 2.0, 3.0, 4.0, 5.0, 6.0, 8.0, 9.0, 10.0]

Lista Ordenada
[1.0, 2.0, 2.0, 3.0, 4.0, 5.0, 6.0, 8.0, 9.0, 10.0]
Tiempo transcurrido: 35.84143352508545 segundos
PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]

Lista Ordenada
[1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]

Tiempo transcurrido: 16.921122312545776 segundos
PS D:\Analisis y diseño de algoritmos>
```

2. Burbuja

```
Burbuja

Lista
[11.0, 12.0, 13.0, 14.0, 15.0, 16.0, 17.0, 18.0, 19.0, 20.0]

Lista Ordenada
[11.0, 12.0, 13.0, 14.0, 15.0, 16.0, 17.0, 18.0, 19.0, 20.0]
Tiempo transcurrido: 27.915559768676758 segundos
PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [11.0, 12.0, 15.0, 16.0, 17.0, 18.0, 19.0, 20.0, 21.0, 22.0, 23.0, 24.0, 25.0,

Lista Ordenada
[11.0, 12.0, 15.0, 16.0, 17.0, 18.0, 19.0, 20.0, 21.0, 22.0, 23.0, 24.0, 25.0, 2

Tiempo transcurrido: 40.74427676200867 segundos
PS D:\Analisis y diseño de algoritmos>
```

3. Burbuja

```
Burbuja

Lista
[111.0, 112.0, 113.0, 113.0, 114.0, 115.0, 116.0, 117.0, 118.0, 119.0, 120.0, 121

Lista Ordenada
[111.0, 112.0, 113.0, 113.0, 114.0, 115.0, 116.0, 117.0, 118.0, 119.0, 120.0, 121
Tiempo transcurrido: 54.93109965324402 segundos
PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [10.0, 20.0, 30.0, 40.0, 50.0, 60.0, 70.0, 80.0, 90.0, 100.0]

Lista Ordenada
[10.0, 20.0, 30.0, 40.0, 50.0, 60.0, 70.0, 80.0, 90.0, 100.0]

Tiempo transcurrido: 40.16357421875 segundos
PS D:\Analisis y diseño de algoritmos>
```

- Caso promedio: $O(n^2)$

1. Burbuja

```
Burbuja

Lista
[2.0, 3.0, 45.0, 79.0, 223.0, 12.0, 90.0]

Lista Ordenada
[2.0, 3.0, 12.0, 45.0, 79.0, 90.0, 223.0]
Tiempo transcurrido: 17.952050924301147 segundos
PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [23.0, 57.0, 2.0, 4.0, 46.0, 20.0, 2.0, 12.0, 1.0, 5.0]

Lista Ordenada
[1.0, 2.0, 2.0, 4.0, 5.0, 12.0, 20.0, 23.0, 46.0, 57.0]

Tiempo transcurrido: 19.481131076812744 segundos
PS D:\Analisis y diseño de algoritmos>
```

2. Burbuja

```
 Burbuja

 Lista
 [7.0, 2.0, 1.0, 3.0, 8.0]

 Lista Ordenada
 [1.0, 2.0, 3.0, 7.0, 8.0]
 Tiempo transcurrido: 31.67137050628662 segundos
 PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
 Burbuja Optimizada
 Lista
  [9.0, 2.0, 3.0, 4.0, 5.0, 7.0, 5.0, 2.0, 1.0, 2.0]

 Lista Ordenada
 [1.0, 2.0, 2.0, 2.0, 3.0, 4.0, 5.0, 5.0, 7.0, 9.0]

 Tiempo transcurrido: 25.213776111602783 segundos
 PS D:\Analisis y diseño de algoritmos>
```

3. Burbuja

```
Burbuja

Lista
[234.0, 54.0, 39.0, 239.0, 2459.0, 34.0, 239.0, 3404.0, 30.0, 4.0]

Lista Ordenada
[4.0, 30.0, 34.0, 39.0, 54.0, 234.0, 239.0, 239.0, 2459.0, 3404.0]
Tiempo transcurrido: 18.95153307914734 segundos
PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [2.0, 47.0, 437.0, 222.0, 44.0, 24.0, 11.0, 23.0, 1.0, 38.0, 40.0, 34.0, 50.0,

Lista Ordenada
[1.0, 2.0, 11.0, 21.0, 23.0, 23.0, 24.0, 34.0, 35.0, 38.0, 40.0, 44.0, 47.0, 50.

Tiempo transcurrido: 37.445842027664185 segundos
PS D:\Analisis y diseño de algoritmos>
```

- Peor caso: $O(n^2)$

1. Burbuja

```
Burbuja

Lista
[10.0, 9.0, 8.0, 7.0, 6.0, 5.0, 4.0, 3.0, 2.0, 1.0]

Lista Ordenada
[1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]
Tiempo transcurrido: 51.71857452392578 segundos
PS D:\Analisis y diseño de algoritmos>
```

Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [10.0, 9.0, 8.0, 7.0, 6.0, 5.0, 4.0, 3.0, 2.0, 1.0]

Lista Ordenada
[1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]

Tiempo transcurrido: 14.960274457931519 segundos
PS D:\Analisis y diseño de algoritmos>
```

2. Burbuja

```
Burbuja

Lista
[1000.0, 900.0, 899.0, 799.0, 788.0, 699.0, 698.0, 500.0, 400.0, 100.0]

Lista Ordenada
[100.0, 400.0, 500.0, 698.0, 699.0, 788.0, 799.0, 899.0, 900.0, 1000.0]
Tiempo transcurrido: 70.09897446632385 segundos
PS D:\Analisis y diseño de algoritmos>
```

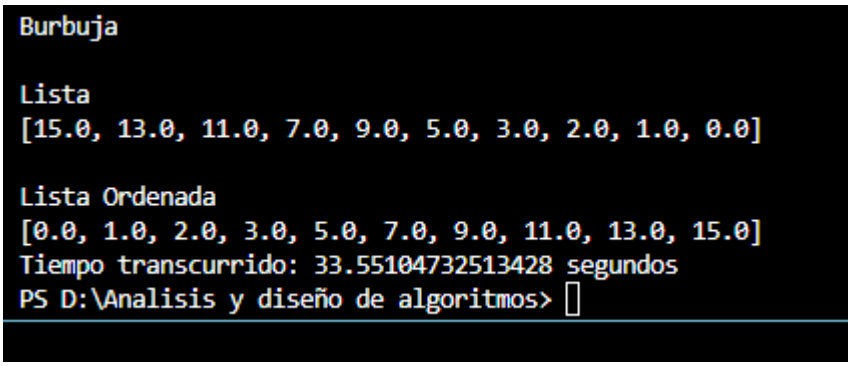
Burbujas Optimizada

```
Burbuja Optimizada
Lista
 [1000.0, 900.0, 800.0, 700.0, 600.0, 500.0, 400.0, 300.0, 200.0, 100.0]

Lista Ordenada
[100.0, 200.0, 300.0, 400.0, 500.0, 600.0, 700.0, 800.0, 900.0, 1000.0]

Tiempo transcurrido: 29.93366026878357 segundos
PS D:\Analisis y diseño de algoritmos>
```

3. Burbuja



Burbujas Optimizada