**PRACTICA Nro. 1: COMPLEJIDAD DE ALGORITMOS**

**Utilizando el software Concorde con un timer de 60 segundos, llene el cuadro siguiente**

<http://www.math.uwaterloo.ca/tsp/concorde/downloads/downloads.htm>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Instancias (Nro. De ciudades)** | **Método Exacto (Busca el óptimo)**  **cutting-plane method** | | | **Método: Heurísticas**  **(Buscan una buena solución en un tiempo razonable)** | | |
| **Greedy** | **Nearest neighbor** | **Random** |
| **Optimo**  **(Si/no)** | **Valor objetivo** | **Tiempo** | **Valor objetivo** | **Valor objetivo** | **Valor objetivo** |
| **50** |  |  |  |  |  |  |
| **100** |  |  |  |  |  |  |
| **300** |  |  |  |  |  |  |
| **400** |  |  |  |  |  |  |
| **500** |  |  |  |  |  |  |
| **600** |  |  |  |  |  |  |
| **700** |  |  |  |  |  |  |
| **800** |  |  |  |  |  |  |
| **900** |  |  |  |  |  |  |
| **1000** |  |  |  |  |  |  |
| **1200** |  |  |  |  |  |  |
| **1400** |  |  |  |  |  |  |
| **1600** |  |  |  |  |  |  |
| **1800** |  |  |  |  |  |  |
| **2000** |  |  |  |  |  |  |
| **2500** |  |  |  |  |  |  |
| **3000** |  |  |  |  |  |  |
| **4000** |  |  |  |  |  |  |
| **5000** |  |  |  |  |  |  |
| **6000** |  |  |  |  |  |  |

1. Para el método exacto haga una gráfica Instancias x Tiempo. Discutir la gráfica.
2. Haga un solo gráfico Instancias x Valor objetivo los cuatro métodos. Discutir la gráfica

**PRACTICA Nro. 2: COMPLEJIDAD DE ALGORITMOS**

**Programe los métodos de ordenamiento:**

**a) Método de la burbuja**

**b) Método de ordenamiento por** **inserción**

**Con un timer máximo de 120 segundos, genere listas de números en forma aleatoria y llene el cuadro siguiente**

|  |  |  |
| --- | --- | --- |
| **Instancias (Lista de números)** | **Tiempo de ordenamiento** | |
| **Método de la burbuja** | **Método de ordenamiento por** **inserción** |
| **100** |  |  |
| **200** |  |  |
| **600** |  |  |
| **1000** |  |  |
| **2000** |  |  |
| **4000** |  |  |
| **6000** |  |  |
| **8000** |  |  |
| **10000** |  |  |
| **20000** |  |  |
| **30000** |  |  |
| **40000** |  |  |
| **50000** |  |  |
| **10000** |  |  |

Haga un solo gráfico Instancias x Tiempo de ordenamiento para los dos métodos. Discutir la gráfica

Codigo en Python

Programa1.py

def ordenar\_burbuja(L):

for i in range(1,len(L)):

for j in range(0, len(L)-i):

if L[j] > L[j+1]:

temp=L[j]

L[j]=L[j+1]

L[j+1]=temp

return L

programa2.py

*def ordenar\_insertando(L):*

*for j in range(1,len(L)):*

*i=j-1*

*temp=L[j]*

*while i > -1 and L[i] > temp:*

*L[i+1]=L[i]*

*i=i-1*

*L[i+1]=temp*

*return L*

Ejemplo de como generar la lista de números aleatorios

>>> G1=random.sample(range(10\*\*3),10\*\*3)

>>> G2=random.sample(range(10\*\*4),10\*\*4)

>>> G3=random.sample(range(10\*\*5),10\*\*5)