Laboratorio de la semana 2

Descripción de la actividad

El objetivo de este laboratorio es la creación de una librería, llamada *ordenamiento.py (http://ordenamiento.py)*, que contenga algoritmos de ordenamientos simples. Para ello debe implementar los siguientes algoritmos, siguiendo el pseudo código usado en las referencias.

1. Insertion-Sort, en página 18 de **[C]**
2. Selection-Sort, en las páginas 256-257 **[A]**
3. Shell-Sort, en la página 290 **[A]**
4. Bubble-Sort, en página 40 de **[C]**

Se quiere que implemente un programa, llamado *prueba_ord.py* para probar la librería *ordenamiento.py (http://ordenamiento.py)*. El programa debe ejecutar varias pruebas sobre todos los algoritmos de ordenamiento de la librería. Las pruebas tienen diferentes secuencias que van a recibir los algoritmos de ordenamiento y se describen como sigue:

1. Secuencia de números reales en el intervalo [0, 1] generados aleatoriamente
2. Secuencie de números enteros en el intervalo [0, N] generados aleatoriamente, donde N es el tamaño de la secuencia a ordenar.
3. Una secuencia de tipo 1, pero en donde todos los elementos están ordenados
4. Una secuencia de tipo 2, pero en donde todos los elementos están ordenados
5. Una secuencia de tipo 1, pero en donde todos los elementos están ordenados de forma inversa
6. Una secuencia de tipo 2, pero en donde todos los elementos están ordenados de forma inversa

Para ordenar las secuencias de tipo 3 y 4, que van a ser las entradas de los algoritmos de ordenamiento, puede usar los métodos **sort** y **sorted** de Python.

El usuario puede seleccionar la pueba a aplicar y el número de veces, o intentos, que se va aplicar esa prueba sobre los algoritmos de ordenamiento. También el usuario puede indicar el tamaño de la secuencia de entrada. El resultado del programa es el tiempo promedio de ejecución de cada algoritmo en ordenar una secuencia dada de un tipo específico, junto con la desviación estándar.

La ejecución de *prueba_ord.py* se hace por medio de la siguiente línea de comando:

```
>./prueba_ord.py [-n <num_elementos>] [-i <num_intentos>] [-t
<pueba_a_ejecutar>]
```

Todos los argumentos de la línea de comandos son opcionales. Los valores por de defecto son:

- **-n** 1000, es decir, la secuencia por defecto tienen mil elementos
- **-i** 3, es decir, se ejecutará tres veces sobre los algoritmos de ordenamiento la prueba seleccionada
- **-t** 1, es decir, el tipo de secuencia de entrada, es la generada por el tipo 1

Ejemplos de llamadas válidas del programa

```
>./prueba_ord.py

>./prueba_ord.py -n 2000

>./prueba_ord.py -n 1500 -t 6

>./prueba_ord.py -n 1500 -t 6 -i 5
```

Para manejar los argumentos de entrada del programa, debe hacer uso del módulo argparse (https://docs.python.org/dev/library/argparse.html) o del módulo getopt (https://docs.python.org/dev/library/getopt.html)

La salida del programa es tiempo promedio de los cuatro algoritmos de ordenamiento, junto con la desviación estándar.

Condiciones de entrega

Debe hacer la entrega de la versión final del código del laboratorio, en GitHub, antes de las 11:50 pm del día domingo 19 de enero de 2020.

Referencias

- T. Cormen, C. Leirserson, R. Rivest. y C. Stein. *Introduction to Algorithms*. Tercera edición. McGraw Hill, 2009. **[C]**
- A. Aho, J. Hopcroft y J. Ullman. *Estructuras de Datos y Algoritmos*. Addison-Wesley, 1998. **[A]**