Karol José Gutiérrez Suárez
A01024536

Radix Sort y Heap Sort

Manual de Usuario

0. El siguiente manual explica operaciones de ordenamiento sobre un arreglo usando los algoritmos de Radix Sort y Heap Sort y que son ejecutadas a través de una clase en el lenguaje de programación Java. Para utilizar esta clase es necesario tener Java instalado en el equipo y colocar el archivo de texto con los datos de entrada dentro de la misma carpeta en donde se encuentre el archivo Java que acompaña a este manual. Esta API fue construida y probada utilizando Eclipse IDE.

Se mostrarán las funciones de las clases y algunos de sus ejemplos de uso.

1. **La clase Arreglo**
   Esta clase es la básica primaria que guardará todos las funciones y el arreglo de nuestra implementación.
   Dentro de la función main que se utilice, será necesario declarar un objeto de la clase arreglo sobre el cual se realizarán las operaciones.
   Ejemplo:
   Arreglo miArreglo=new Arreglo();
2. **Función void lecturaDatos(string archivo)**
   Esta función es la que recibe los datos provenientes de un archivo de texto que se encuentre en la misma carpeta en la que se encuentra el archivo java.
   miArreglo.lecturaDatos("cualquierNombre.txt");
   El archivo deberá contener un entero n en la primera línea, y en las siguientes n líneas deberá tener los n elementos del arreglo. Dichos elementos del arreglo deberán ser enteros y en caso de utilizar RadixSort tendrán que ser enteros no negativos.
3. **Función void imprimeArreglo();**
   En cualquier momento se puede llamar a está función que imprime los elementos del arreglo en el orden en el que se encuentren.
   Ejemplo:
   miArreglo.imprimeAreglo();
   Salida:

   ```
   El arreglo tiene 20 elementos, que son:
   72 0 36 27 46 46 21 18 82 -8 83 42 21 -12 23 50 10 67 49 0
   ```
4. **Función void heapSort();**
   Ordena el arreglo usando el algoritmo de Heap Sort. Cuando se llama, imprime una nueva línea que dice "`Ordenando arreglo con Heap Sort...`".
   Ejemplo:
   miArreglo.heapSort();
   Salida:
   `Ordenando arreglo con Heap Sort...`

5. **Función void radixSort()**

Ordena el arreglo usando el algoritmo de Radix Sort. Cuando se llama, imprime una nueva línea que dice "`Ordenando arreglo con Radix Sort...`".
Ejemplo:

miArreglo.radixSort();

Salida:

```
Ordenando arreglo con Radix Sort...
```

6. **Verificando el ordenamiento**

Una vez que se ordenan los arreglos usando cualquiera de las dos funciones anteriores, es posible verificar que está bien ordenado usando de nuevo la función imprimeArreglo.

Ejemplo:

```
miArreglo.imprimeArreglo();
miArreglo.heapSort();
miArreglo.imprimeArreglo();
```

Salida:

```
El arreglo tiene 20 elementos, que son:
72 0 36 27 46 46 21 18 82 -8 83 42 21 -12 23 50 10 67 49 0
Ordenando arreglo con Heap Sort...
El arreglo tiene 20 elementos, que son:
-12 -8 0 0 10 18 21 21 23 27 36 42 46 46 49 50 67 72 82 83
```

Ejemplos de uso

El siguiente ejemplo forma parte de un mismo programa y ejemplifica lo que le ocurre al arreglo.

```
170    public static void main(String[] args) {
171        Arreglo ar=new Arreglo();
172
173
174        ar.lecturaDatos("prueba.txt");
175        ar.imprimeArreglo();
176        ar.heapSort();
177        ar.imprimeArreglo();
178
179        System.out.println();
180        ar.lecturaDatos("otraprueba.txt");
181        ar.imprimeArreglo();
182        ar.radixSort();
183        ar.imprimeArreglo();
184
185    }
186
```

Karol José Gutiérrez Suárez
A01024536

prueba.txt - Notepad

File Edit Format View Help

```
20
72
0
36
27
46
46
21
18
82
-8
83
42
21
-12
23
50
10
67
49
0
```

```
ar.lecturaDatos("prueba.txt");
ar.imprimeArreglo();
ar.heapSort();
ar.imprimeArreglo();
```

Problems Javadoc Declaration Console Coverage

```
<terminated> Arreglo (1) [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\ja
El arreglo tiene 20 elementos, que son:
72 0 36 27 46 46 21 18 82 -8 83 42 21 -12 23 50 10 67 49 0

Ordenando arreglo con Heap Sort...
El arreglo tiene 20 elementos, que son:
-12 -8 0 0 10 18 21 21 23 27 36 42 46 46 49 50 67 72 82 83
```

otraprueba.txt - Notepad

File Edit Format View Help

```
12
4
67
21
45
0
32
27
66
87
31
60
2
```

```
System.out.println();
ar.lecturaDatos("otraprueba.txt");
ar.imprimeArreglo();
ar.radixSort();
ar.imprimeArreglo();
```

```
El arreglo tiene 12 elementos, que son:
4 67 21 45 0 32 27 66 87 31 60 2

Ordenando arreglo con Radix Sort...
El arreglo tiene 12 elementos, que son:
0 2 4 21 27 31 32 45 60 66 67 87
```