ESTRUCTURAS DE DATOS Y ALGORITMOS II **2021-1**

PRACTICA # 2: ALGORITMOS DE ORDENAMIENTO PARTE 2

OBJETIVO: El estudiante identificará la estructura de los algoritmos de ordenamiento BubbleSort, QuickSort y HeapSort.

Objetivo de clase: El estudiante observará la importancia del orden de complejidad aplicado en algoritmos de ordenamiento, conocerá diferentes formas de implementar Quicksort y desarrollará habilidades básicas de programación orientada a objetos

ACTIVIDADES PARA EL DESARROLLO DE LA PRÁCTICA

Ejercicio 1. Agregando Ordenamientos

Añade a tu biblioteca "ordenamientos.h" las siguientes funciones correspondientes a los algoritmos de QuickSort, Bubble sort y Heap Sort (heapSort se encuentra en la plataforma EDUCAFI)

```
int partition (int arr[], int low, int high){
    int pivot = arr[high];
printf("Pivote: %d    ",pivot);
  int j,i = (low - 1);
    for (j = low; j <= high- 1; j++){
        if (arr[j] <= pivot){
                i++;
                swap(&arr[i], &arr[j]);
            }
        }
        swap(&arr[i + 1], &arr[high]);
return (i + 1);
}

void quickSort(int arr[], int low, int high){
if (low < high){
    int pi = partition(arr, low, high);
          quickSort(arr, low, pi - 1);
          quickSort(arr, pi + 1, high);
    }
}

void bubbleSort(int a[], int size){
  int i,j,n;
  n= size;
  for(i=n-1;i>0;i--){
    for(j=0; j<i; j++){
      if(a[j]>a[j+1])
        swap(&a[j], &a[j+1]);
    }
  }
}
```

- Analiza cuidadosamente el algoritmo de QuickSort e indica si se trata de alguna de las implementaciones vistas en clase o es una diferente, deberás incluir evidencia incontrovertible para sustentar tu afirmación
- Agrega al algoritmo de BubbleSort, las instrucciones necesarias para que el algoritmo se detenga cuando este ordenado el arreglo
- Describe la función HEAPIFY del algoritmo de Heap Sort (el algoritmo HeapSort lo encontrarás en Educafi) e indica si la construcción del HEAP es TopDown o Bottom UP

2.- Verificando el funcionamiento

Al igual que en la práctica anterior, deberás crear un arreglo de 10 elementos que se llenará de manera aleatoria, realizar el ordenamiento con los diferentes algoritmos y mostrar el arreglo original, el estado del arreglo al final de cada iteración y el arreglo ordenado.

3.- Complejidad Computacional

Agrega en los algoritmos de ordenamiento, las instrucciones necesarias para contabilizar el número de operaciones relevantes (comparaciones, intercambios, etc), realiza pruebas con arreglos de los mismos tamaños de la práctica 1. (con ayuda de gráficas),

4. Quicksort en Java y practicando la orientación a objetos

Con ayuda de NetBeans crea un nuevo proyecto "Practica2[ApellidoPaternoyNombre]", agrega una clase llamada Quicksort con el siguiente código

```
int partition(int arr[], int low, int high){
        int pivot = arr[high];
        int i = (low-1);
        for (int j=low; j<high; j++){
            if (arr[j] <= pivot)
            {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }

        int temp = arr[i+1];
        arr[i+1] = arr[high];
        arr[high] = temp;
        return i+1;
    }

    void sort(int arr[], int low, int high){
        if (low < high){
            int pi = partition(arr, low, high);
            sort(arr, low, pi-1);
            sort(arr, pi+1, high);
        }
    }
```

Agrega al proyecto una nueva clase llamada utilidades con los siguientes elementos

```
static void printArray(int arr[]){
  int n = arr.length;
  for (int i=0; i<n; ++i){
    System.out.print(arr[i]+" ");
    System.out.println();
  }
}
```

En la clase principal de tu proyecto (la que tiene la función main) agrega las instrucciones necesarias para crear un objeto de tipo "quickSort" y probar el ordenamiento

```
public static void main(String args[]){
  int arr[] = {87,4,32,15,8,12,10,30,22};
  int n = arr.length;
  //aquí crea un objeto
  //aquí manda llamar el método de ordenamiento
  printArray(arr);
}
```

Realiza un análisis del programa y sobre todo indica los elementos de la programación en Java

5.- Escribe tus conclusiones de la práctica.