**Trabajo practico nro 7**

|  |  |  |
| --- | --- | --- |
| Logo Departamento 2.jpg | **Asignatura: Laboratorio I** | |
|  | |
| **Cursado:**Primer Trimestre | **Horas** **semanales**: |
|  | **Horas semestrales:**  *Cantidad estimada de horas semestrales/anuales.* |
| **Carrera**: *Tecnicatura Universitaria en Programación* | **Nivel (Año):** |
| **Ciclo Lectivo: 2023** |

1. **Tema:**

**Algoritmos de ordenación**

1. **Enunciados:**

**Empezaremos por unos ejercicios básicos de programas Java con estructura secuencial, es decir, en estos programas no hay instrucciones condicionales ni repetitivas. En la mayoría de ellos las operaciones a realizar son: lectura de datos por teclado, realizar alguna operación con esos datos y mostrar resultados por pantalla.**

**Recomendaciones: Leer la teoría, investigar, consultar si existen dudas y realizar el trabajo en clases. La entrega de este trabajo practico es individual.**

1. Ordenar un array de números enteros de menor a mayor.

import java.util.Scanner;  
import java.util.Arrays;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 int[] array={2, 7, 3, 10, 5};  
 System.*out*.println("Version cruda: ");  
 for (int j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array);  
 System.*out*.println("Version ordenada de menor a mayor");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
  
 }  
}

1. Ordenar un array de números enteros de mayor a menor.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 Integer[] array={2, 7, 3, 10, 5};  
 System.*out*.println("Version cruda: ");  
 for (int j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array, Collections.*reverseOrder*());  
 System.*out*.println("Version ordenada de mayor a menor");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
  
 }  
}

1. Ordenar un array de números flotantes de menor a mayor.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 float[] array={2.0f, 3.4f, 3.5f, 3.9f, 5.6f, 2.4f};  
 System.*out*.println("Version cruda: ");  
 for (float j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array);  
 System.*out*.println("Version ordenada de menor a mayor");  
 for(float i: array){  
 System.*out*.println(i);  
 }  
  
 }  
}

1. Ordenar un array de números flotantes de mayor a menor.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 Float[] array={2.0f, 3.4f, 3.5f, 3.9f, 5.6f, 2.4f};  
 System.*out*.println("Version cruda: ");  
 for (float j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array, Collections.*reverseOrder*());  
 System.*out*.println("Version ordenada de mayor a menor");  
 for(float i: array){  
 System.*out*.println(i);  
 }  
  
 }  
}

1. Ordenar un array de cadenas de texto en orden alfabético.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 String[] array={"Naza", "Alvaro", "Zack", "Valeria", "Rodrigo", "Camila"};  
 System.*out*.println("Version cruda: ");  
 for (String j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array);  
 System.*out*.println("Version ordenada alfabeticamente");  
 for(String i: array){  
 System.*out*.println(i);  
 }  
  
 }  
}

1. Ordenar un array de cadenas de texto en orden inverso.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 String[] array={"Naza", "Alvaro", "Zack", "Valeria", "Rodrigo", "Camila"};  
 System.*out*.println("Version cruda: ");  
 for (String j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array, Collections.*reverseOrder*());  
 System.*out*.println("Version ordenada al inverso del alfabeto");  
 for(String i: array){  
 System.*out*.println(i);  
 }  
  
 }  
}

1. Ordenar un array de objetos de una clase personalizada.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 Iden[] array;  
 array= new Iden[5];  
 array[0]= new Iden("Naza", 18);  
 array[1]= new Iden("Rodrigo", 19);  
 array[2]= new Iden("Thomas", 18);  
 array[3]= new Iden("Valen", 8);  
 array[4]= new Iden("Paulo", 46);  
 System.*out*.println("Version cruda: ");  
 for (Iden j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array);  
 System.*out*.println("Version ordenada de menor a mayor segun el atributo edad");  
 for(Iden i: array){  
 System.*out*.println(i);  
 }  
 }  
}  
class Iden implements Comparable<Iden>{  
  
 private String name;  
 private int age;  
 public Iden(String name, int age){  
 this.age=age;  
 this.name=name;  
 }  
 public int compareTo(Iden other){  
 return Integer.*compare*(this.age, other.age);  
 }  
 public String toString(){  
 return "Nombre: "+name+", Edad: "+age;  
 }  
 public String getName(){  
 return name;  
 }  
 class NameComparator implements Comparator<Iden>{  
 public int compare(Iden o1, Iden o2){  
 return o1.getName().compareTo(o2.getName());  
 }  
 }  
}

1. Ordenar un array de objetos de una clase personalizada por un atributo específico.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 Iden[] array;  
 array= new Iden[5];  
 array[0]= new Iden("Naza", 18);  
 array[1]= new Iden("Rodrigo", 19);  
 array[2]= new Iden("Thomas", 18);  
 array[3]= new Iden("Valen", 8);  
 array[4]= new Iden("Paulo", 46);  
 System.*out*.println("Version cruda: ");  
 for (Iden j : array) {  
 System.*out*.println(j);  
 }  
 Arrays.*sort*(array, new NameComparator());  
 System.*out*.println("Version ordenada alfabeticamente segun el atributo name");  
 for(Iden i: array){  
 System.*out*.println(i);  
 }  
 }  
}  
class Iden implements Comparable<Iden>{  
  
 private String name;  
 private int age;  
 public Iden(String name, int age){  
 this.age=age;  
 this.name=name;  
 }  
 public int compareTo(Iden other){  
 return Integer.*compare*(this.age, other.age);  
 }  
 public String toString(){  
 return "Nombre: "+name+", Edad: "+age;  
 }  
 public String getName(){  
 return name;  
 }  
 }  
class NameComparator implements Comparator<Iden>{  
 public int compare(Iden o1, Iden o2){  
 return o1.getName().compareTo(o2.getName());  
 }  
}

1. Ordenar un array de números enteros utilizando el algoritmo de ordenamiento burbuja.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 BubbleSort Bubble= new BubbleSort();  
 Scanner sc = new Scanner(System.*in*);;  
 int[] array={1, 4, 7, 3, 8};  
 System.*out*.println("Forma normal");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 Bubble.array(array);  
 System.*out*.println("Forma ascendente mediante algortimo de burbuja");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 }  
 }  
class BubbleSort{  
 void array(int[] arraylocal){  
 for(int i=0; i<arraylocal.length-1;++i){  
 for(int j=0; j<arraylocal.length-1-i;++j){  
 if(arraylocal[j]>arraylocal[j+1]){  
 int tempor=arraylocal[j];  
 arraylocal[j]=arraylocal[j+1];  
 arraylocal[j+1]=tempor;  
 }  
 }  
 }  
 }  
}

1. Ordenar un array de números enteros utilizando el algoritmo de ordenamiento por selección.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 SelectSort Sel= new SelectSort();  
 Scanner sc = new Scanner(System.*in*);;  
 int[] array={1, 4, 7, 3, 8};  
 System.*out*.println("Forma normal");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 Sel.array(array);  
 System.*out*.println("Forma ascendente mediante algoritmo de seleccion");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 }  
 }  
class SelectSort {  
 void array(int[] arraylocal) {  
 int n = arraylocal.length;  
  
 for (int i = 0; i < n - 1; i++) {  
 int minIndex = i;  
 for (int j = i + 1; j < n; j++) {  
 if (arraylocal[j] < arraylocal[minIndex]) {  
 minIndex = j;  
 }  
 }  
 int temp = arraylocal[minIndex];  
 arraylocal[minIndex] = arraylocal[i];  
 arraylocal[i] = temp;  
 }  
 }  
}

1. Ordenar un array de números enteros utilizando el algoritmo de ordenamiento por inserción.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 DirIn ins= new DirIn();  
 Scanner sc = new Scanner(System.*in*);;  
 int[] array={1, 4, 7, 3, 8};  
 System.*out*.println("Forma normal");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 ins.array(array);  
 System.*out*.println("Forma ascendente mediante algoritmo de insercion directa");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 }  
 }  
 class DirIn{  
 void array(int[] arraylocal){  
 int aux, pbef, j;  
 for(int i=1;i<arraylocal.length;i++){  
 aux=arraylocal[i];  
 pbef=i-1;  
 while(pbef>=0 && aux<arraylocal[pbef]){  
 arraylocal[pbef+1]=arraylocal[pbef];  
 pbef--;  
 }  
 arraylocal[pbef+1]=aux;  
 }  
 }  
 }

1. Ordenar un array de números enteros utilizando el algoritmo de ordenamiento por mezcla.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 int[] array={1, 4, 7, 3, 8};  
 System.*out*.println("Forma normal");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 MergeSort.*sort*(array);  
 System.*out*.println("Forma ascendente mediante algoritmo de mezcla");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 }  
 }  
  
class MergeSort {  
 public static void sort(int[] array) {  
 *mergeSort*(array, 0, array.length - 1);  
 }  
  
 private static void mergeSort(int[] array, int left, int right) {  
 if (left < right) {  
 int mid = left + (right - left) / 2;  
 *mergeSort*(array, left, mid);  
 *mergeSort*(array, mid + 1, right);  
 *merge*(array, left, mid, right);  
 }  
 }  
  
 private static void merge(int[] array, int left, int mid, int right) {  
 int leftLength = mid - left + 1;  
 int rightLength = right - mid;  
 int[] leftArray = new int[leftLength];  
 int[] rightArray = new int[rightLength];  
 for (int i = 0; i < leftLength; i++) {  
 leftArray[i] = array[left + i];  
 }  
 for (int j = 0; j < rightLength; j++) {  
 rightArray[j] = array[mid + 1 + j];  
 }  
 int i = 0, j = 0;  
 int k = left;  
 while (i < leftLength && j < rightLength) {  
 if (leftArray[i] <= rightArray[j]) {  
 array[k++] = leftArray[i++];  
 } else {  
 array[k++] = rightArray[j++];  
 }  
 }  
 while (i < leftLength) {  
 array[k++] = leftArray[i++];  
 }  
 while (j < rightLength) {  
 array[k++] = rightArray[j++];  
 }  
 }  
}

1. Ordenar un array de números enteros utilizando el algoritmo de ordenamiento shell.

import java.util.Scanner;  
import java.util.Arrays;  
import java.util.Collections;  
import java.util.Comparator;  
public class Main {  
 public static void main(String[] args){  
 Scanner sc = new Scanner(System.*in*);;  
 int[] array={1, 4, 7, 3, 8};  
 System.*out*.println("Forma normal");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 ShellSort.*sort*(array);  
 System.*out*.println("Forma ascendente mediante algoritmo de shell");  
 for(int i: array){  
 System.*out*.println(i);  
 }  
 }  
 }  
class ShellSort {  
 public static void sort(int[] array) {  
 int n = array.length;  
 int gap = n / 2;  
 while (gap > 0) {  
 for (int i = gap; i < n; i++) {  
 int temp = array[i];  
 int j = i;  
 while (j >= gap && array[j - gap] > temp) {  
 array[j] = array[j - gap];  
 j -= gap;  
 }  
 array[j] = temp;  
 }  
 gap /= 2;  
 }  
 }

1. Codifique un algoritmo que permita cargar un array de una dimensión de 20 elementos enteros. Finalizada la carga de los 20 elementos el programa debe solicitar como se desea ordenar el array ASCENDENTE O DESCENDENTE y que método de ordenamiento aplicar (inserción, burbuja, selección), según las opciones elegidas aplique el algoritmos fundamental de ordenamiento que corresponda. Muestre por pantalla el array original desordenado y su resultante ordenado según las opciones elegidas.
2. import java.util.Scanner;  
   import java.util.Arrays;  
   import java.util.Collections;  
   import java.util.Comparator;  
   public class Main {  
    public static void main(String[] args){  
    Scanner sc = new Scanner(System.*in*);  
    SelectSort sel=new SelectSort();  
    BubbleSort bub= new BubbleSort();  
    DirIn dir= new DirIn();  
    int choice, des;  
    int[] array={1, 9, 4, 2, 5, 7, 8, 6, 11, 16, 12, 13, 14, 18, 21, 17, 22, 24, 26};  
    System.*out*.println("Cual forma de ordenamiento quiere?");  
    System.*out*.println("1) Seleccion");  
    System.*out*.println("2) Burbuja");  
    System.*out*.println("3) Insercion");  
    choice= sc.nextInt();  
    if (choice==1){  
    System.*out*.println("Ascendente o descendente? (0 para ascendente, 1 para descendente)");  
    des=sc.nextInt();  
    if(des==0){  
    System.*out*.println("Forma normal");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    sel.array(array);  
    System.*out*.println("Forma ascendente mediante algoritmo de seleccion");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    }else if (des==1){  
    System.*out*.println("Forma normal");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    sel.arrayinv(array);  
    System.*out*.println("Forma descendente mediante algoritmo de seleccion");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    }  
    }else if(choice==2){  
    System.*out*.println("Ascendente o descendente? (0 para ascendente, 1 para descendente)");  
    des=sc.nextInt();  
    if(des==0){  
    System.*out*.println("Forma normal");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    bub.array(array);  
    System.*out*.println("Forma ascendente mediante algoritmo de burbuja");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    } else if (des==1){  
    System.*out*.println("Forma normal");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    bub.arrayinv(array);  
    System.*out*.println("Forma descendente mediante algoritmo de burbuja");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    }  
    } else if (choice==3){  
    System.*out*.println("Ascendente o descendente? (0 para ascendente, 1 para descendente)");  
    des=sc.nextInt();  
    if(des==0){  
    System.*out*.println("Forma normal");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    dir.array(array);  
    System.*out*.println("Forma ascendente mediante algoritmo de insercion");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    } else if (des==1){  
    System.*out*.println("Forma normal");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    dir.arrayinv(array);  
    System.*out*.println("Forma descendente mediante algoritmo de insercion");  
    for(int i: array){  
    System.*out*.println(i);  
    }  
    }  
    }  
    }  
    }

class DirIn{  
 void array(int[] arraylocal){  
 int aux, pbef, j;  
 for(int i=1;i<arraylocal.length;i++){  
 aux=arraylocal[i];  
 pbef=i-1;  
 while(pbef>=0 && aux<arraylocal[pbef]){  
 arraylocal[pbef+1]=arraylocal[pbef];  
 pbef--;  
 }  
 arraylocal[pbef+1]=aux;  
 }  
 }  
 void arrayinv(int[] arraylocal){  
 int aux, pbef, j;  
 for(int i=1;i<arraylocal.length;i++){  
 aux=arraylocal[i];  
 pbef=i-1;  
 while(pbef>=0 && aux>arraylocal[pbef]){  
 arraylocal[pbef+1]=arraylocal[pbef];  
 pbef--;  
 }  
 arraylocal[pbef+1]=aux;  
 }  
 }  
 }  
class SelectSort {  
 void array(int[] arraylocal) {  
 int n = arraylocal.length;  
  
 for (int i = 0; i < n - 1; i++) {  
 int minIndex = i;  
 for (int j = i + 1; j < n; j++) {  
 if (arraylocal[j] < arraylocal[minIndex]) {  
 minIndex = j;  
 }  
 }  
 int temp = arraylocal[minIndex];  
 arraylocal[minIndex] = arraylocal[i];  
 arraylocal[i] = temp;  
 }  
 }  
 void arrayinv(int[] arraylocal) {  
 int n = arraylocal.length;  
  
 for (int i = 0; i < n - 1; i++) {  
 int minIndex = i;  
 for (int j = i + 1; j < n; j++) {  
 if (arraylocal[j] > arraylocal[minIndex]) {  
 minIndex = j;  
 }  
 }  
 int temp = arraylocal[minIndex];  
 arraylocal[minIndex] = arraylocal[i];  
 arraylocal[i] = temp;  
 }  
 }  
}  
class BubbleSort{  
 void array(int[] arraylocal){  
 for(int i=0; i<arraylocal.length-1;++i){  
 for(int j=0; j<arraylocal.length-1-i;++j){  
 if(arraylocal[j]>arraylocal[j+1]){  
 int tempor=arraylocal[j];  
 arraylocal[j]=arraylocal[j+1];  
 arraylocal[j+1]=tempor;  
 }  
 }  
 }  
 }  
 void arrayinv(int[] arraylocal){  
 for(int i=0; i<arraylocal.length-1;++i){  
 for(int j=0; j<arraylocal.length-1-i;++j){  
 if(arraylocal[j]<arraylocal[j+1]){  
 int tempor=arraylocal[j];  
 arraylocal[j]=arraylocal[j+1];  
 arraylocal[j+1]=tempor;  
 }  
 }  
 }  
 }  
}