**Problemas Propuestos**

**I. Responda en forma concisa y clara**

**a. Cuales son las principales características de una Fila secuencial**

**Respuesta:**

-Cada elemento ocupa un lugar físico contiguo.

-El acceso empieza por el primer elemento.

-Es necesario recorrer todos los elementos antes que n para llegar al elemento n.

-No se puede realizar intersección ni eliminación.

-Reside en almacenamiento secundario(disco).

**b. Cuantas primitivas de acceso existen para tratar filas secuenciales**

**Respuesta:**

Existen 5 primitivas de acceso:

Primer(F), Tomar(F, V), Poner(F, V), Ultimo(F), Marcar(F).

**c. En donde reside una fila secuencial creada**

**Respuesta:**

En la memoria secundaria(disco).

**d. En que consiste la especificación e implementación de soluciones algorítmicas**

**Respuesta:**

-Especificación: Hallar la solución, debe incluir lo necesario para el uso correcto del programa,

y las restricciones para implementar el programa.

-Implementación: Es desarrollar la especificación con sus restricciones.

**e. En que consiste la serialización**

**Respuesta:**

Consiste en codificar un objeto para poder trasmitirlo como una serie de bytes o

un formato legible.

**II. Provea una especificación para:**

**a. Crear un archivo que contenga los números perfectos comprendidos entre M y N**

**Respuesta:**

-Pedir M y N, M < N.

-Crear función numeroPerfecto que compruebe si existe un numero perfecto desde M hasta N, si existe se guarda en el archivo.

-Crear función para guardar numeroPerfecto en archivo.

**b. Crear una fila secuencial para almacenar los N primeros términos de la sucesión 3, 7, 15, 31…**

**Respuesta:**

-Crear una función "sucesión" que sea recursiva y devuelva el termino n de la sucesión, n tiene que ser mayor a 0.

-Crear menú que pida el número de términos de la sucesión.

-Crear función para escribir el archivo con los términos de la sucesión.

**c. Se lee una fila que contiene información, base y altura triángulos. Se quiere hallar el área mayor, área menor y el área promedio**

**Respuesta:**

-Crear objeto triangulo con atributos base y altura.

-Crear función para leer archivo y guardar el valor de la base y altura para cada triangulo.

-Crear función que calcula el área del triángulo.

-Crear función que evalúa si el área es la menor o la mayor hasta el momento y luego sume su valor en una variable suma para luego calcular el promedio.

-Crear función para mostrar

**III. Provea una implementación para solo dos de los ejercicios II.**

**Implementados:**

II. B -> [Sucesion.java](#Sucesion)

II. C -> [Archivo.java](#Archivo), [Area.java](#Area), [Triangulos.java](#Triangulo)

**IV. Se tiene un archivo con números enteros (2453, 123, 5674 …).**

**Construya una solución que verifique cuales son primos.**

**Cuando un número no es primo deberá mostrar la suma de sus dígitos**

**Solución:** [Prob4.java](#Problema4), [Archivo.java](#Archivo)

**V. El ejemplo 15 funciona solo para 3 contactos.**

**Modifique la clase persona, incluya dos atributos más, el que usted desee.**

**Modifique para que se ingresen N contactos.**

**Solución:** [Persona.java](#Persona),[Agenda.java](#Agenda),[Archivo.java](#Archivo)

**VI. Resuelva solo uno de los tres casos, el caso elegido n (n se obtiene aplicando modulo 3 a la suma de los dígitos de su código de alumno, ejemplo si su código es 128, n = Modulo3(1+2+8) = 2) solo resuelva el caso 2**

**----------------------------------**

**|Codigo:15200126 Suma=17 17mod3=2|**

**----------------------------------**

**Caso 0:**

**a) Crear un archivo que contenga información de los alumnos del curso de estructura de datos (código, nombre Examen 1 y Examen 2).**

**b) A partir del archivo creado hallar el mayor, el menor y el promedio de cada uno. Mostrar el mayor y el menor promedio.**

**Caso 1:**

**a. Crear un archivo que contenga información de los pacientes de un hospital (código, nombre, sexo, diagnostico (Sarampión, Covid, etc.) fecha de ingreso y fecha de alta. A partir del archivo creado mostrar los nombres de los pacientes, su diagnóstico, fecha de ingreso y fecha de alta. Mostrar al final del reporte, el nombre del empleado que más gano y el que menos gano.**

**Caso 2:**

**a. Crear un archivo que contenga información de los empleados de una fábrica textil (código, nombre, Sueldo básico, Numero de horas extras)**

**b. A partir del archivo creado hallar el Sueldo bruto (sueldo básico + pago por horas extras, cada hora extra se paga 1.5 hora normal) de cada empleado.**

**Mostrar al final del reporte, el nombre del empleado que más gano y el que menos gano.**

**Solución:**

Caso 2: <Prob6.java>, [Empleado.java](#Empleado), [Archivo.java](#Archivo)

**Class Sucesión.java**

import java.io.\*;

import javax.swing.JOptionPane;

/\*

Crear una fila secuencial para almacenar los N primeros terminos de la sucesión 3, 7, 15, 31…

\*/

class Sucesion {

final String RUTA = "sucesion.dat";

int numeroTerminos;

public Sucesion(int n) {

numeroTerminos = n;

}

public static int sucesion(int n) {

if (n < 1) {

return 3;

} else {

return sucesion(n - 1) \* 2 + 1;

}

}

public void escribirArchivo() throws IOException {

FileWriter fw = new FileWriter(RUTA);

BufferedWriter bw = new BufferedWriter(fw);

PrintWriter sucesion = new PrintWriter(bw);

for (int i = 0; i < numeroTerminos; i++) {

sucesion.print(sucesion(i) + " ");

}

sucesion.close();

System.out.println("El archivo de salida ha sido creado: " + RUTA);

}

public void leerArchivo() throws IOException{

String linea;

FileReader fr = new FileReader(RUTA);

BufferedReader entArch = new BufferedReader(fr);

linea = entArch.readLine();

while (linea != null) {

System.out.println(linea + "\n");

//Se lee una nueva línea

linea = entArch.readLine();

}

entArch.close();

}

}

class SucesionTest {

public static int menu() throws IOException {

int num;

do {

try {

num = Integer.valueOf(JOptionPane.showInputDialog("Ingrese numero de terminos"));

} catch (NumberFormatException e) {

System.out.println("Ingrese un numero valido");

num = -1;

}

} while (num < 0);

return num;

}

public static void main(String[] args) throws IOException {

int numeroTerminos = menu();

Sucesion suc = new Sucesion(numeroTerminos);

suc.escribirArchivo();

suc.leerArchivo();

}

}

**Class Area.java**

import java.io.\*;

import java.util.ArrayList;

/\*

Se lee una fila que contiene información, base y altura triangulos. Se quiere hallar el área mayor, área menor y el área promedio

\*/

public class Area {

static final String RUTA = "baseAltura.dat";

double May = 0, Men, Prom = 0;

int contador = 0;

static ArrayList<Triangulo> triangulos;

public Area() {

triangulos = new ArrayList();

Archivo.leer(RUTA, triangulos);

//guardarArchivo();

// leerArchivo();

May = 0;

Prom = 0;

System.out.println(triangulos.size());

}

void agregarData(ArrayList<Triangulo> data){

Archivo.guardar(RUTA, data);

}

void MayMenProm() {

Men = calcularArea(triangulos.get(0));

for (Triangulo a : triangulos) {

double area = calcularArea(a);

if (area < Men) {

Men = area;

}

if (area > May) {

May = area;

}

Prom += area;

contador++;

System.out.println("Àrea " + contador + " : " + area);

}

mostrar();

}

double calcularArea(Triangulo t) {

return t.getAltura() \* t.getBase() / 2;

}

static void leerArchivo() {

try {

FileInputStream fis = new FileInputStream(RUTA);

ObjectInputStream entrada = new ObjectInputStream(fis);

for (int i = 0;; i++) {

triangulos.add((Triangulo) entrada.readObject());

}

} catch (FileNotFoundException ex) {

System.out.println(ex.getMessage());

} catch (IOException | ClassNotFoundException ex) {

System.out.println("Archivo leido.");

}

}

//

static void guardarArchivo() {

try {

ObjectOutputStream salida = new ObjectOutputStream(new FileOutputStream(RUTA));

salida.writeObject(new Triangulo(2, 6));

salida.writeObject(new Triangulo(5, 6));

salida.writeObject(new Triangulo(5, 8));

salida.writeObject(new Triangulo(7, 8));

salida.close();

} catch (IOException ex) {

System.out.println(ex.getMessage());

}

}

void mostrar() {

System.out.println("El area menor es : " + Men);

System.out.println("El area mayor es : " + May);

System.out.println("El area promedio es : " + Prom);

}

}

class AreaTest {

public static void main(String[] args) {

Area test = new Area();

ArrayList<Triangulo> data = new ArrayList();

data.add(new Triangulo(2, 6));

data.add(new Triangulo(4, 6));

test.agregarData(data);

test.MayMenProm();

}

}

**Class Archivo.java**

import java.io.\*;

import java.util.ArrayList;

class Archivo {

public static void leer(String RUTA, ArrayList a) {

try {

FileInputStream fis = new FileInputStream(RUTA);

ObjectInputStream entrada = new ObjectInputStream(fis);

while (fis.available() > 0) {

a.add(entrada.readObject());

}

} catch (FileNotFoundException ex) {

System.out.println("No existe el archivo: "+RUTA);

} catch (IOException | ClassNotFoundException ex) {

System.out.println("Archivo leido.");

}

}

public static Object leer(String RUTA, Object a) {

try {

FileInputStream fis = new FileInputStream(RUTA);

ObjectInputStream entrada = new ObjectInputStream(fis);

a=entrada.readObject();

} catch (FileNotFoundException ex) {

System.out.println("No existe el archivo: "+RUTA);

} catch (IOException | ClassNotFoundException ex) {

System.out.println("Archivo leido.");

}

return a;

}

public static void guardar(String RUTA, ArrayList a) {

try {

ObjectOutputStream salida = new ObjectOutputStream(new FileOutputStream(RUTA));

for (Object b : a) {

salida.writeObject(b);

}

salida.close();

} catch (IOException ex) {

System.out.println(ex.getMessage());

}

}

public static void guardar(String RUTA, Object a) {

try {

ObjectOutputStream salida = new ObjectOutputStream(new FileOutputStream(RUTA));

salida.writeObject(a);

salida.close();

} catch (IOException ex) {

System.out.println(ex.getMessage());

}

}

}

**Class Triangulo.java**

import java.io.Serializable;

class Triangulo implements Serializable{

private double base;

private double altura;

public Triangulo(double base, double altura) {

this.base = base;

this.altura = altura;

}

public double getAltura() {

return altura;

}

public void setAltura(double altura) {

this.altura = altura;

}

public double getBase() {

return base;

}

public void setBase(double base) {

this.base = base;

}

}

**Class Prob4.java**

import java.util.ArrayList;

class Prob4 {

final static String RUTA = "numeros.dat";

ArrayList<Integer> numeros = new ArrayList();

/\*

Esta Funcion devuelve true si el numero n es Primo.

\*/

int[] cargarNumeros() {

int[] nums;

Archivo.leer(RUTA, numeros);

nums = new int[numeros.size()];

for (int i = 0; i < nums.length; i++) {

nums[i] = numeros.get(i);

}

return nums;

}

void guardarNumeros(int[] num) {

ArrayList<Integer> data = new ArrayList();

for (int a : num) {

data.add(a);

}

Archivo.guardar(RUTA, data);

}

boolean isPrime(int n) {

if (n == 0 || n == 1) {

return false;

}

if (n % 2 == 0) {

return false;

}

for (int i = 3; i <= Math.sqrt(n); i++) {

if (n % i == 0) {

return false;

}

}

return true;

}

int sumaDigitos(int numero) {

int sum = 0;

while (numero > 0) {

sum += numero % 10;

numero /= 10;

}

return sum;

}

}

class Prob4Test {

public static void main(String[] args) {

Prob4 prueba = new Prob4();

int[] a = {2453, 123, 5674, 98, 87, 67, 31, 30, 113, 10};

prueba.guardarNumeros(a);

for (int i : prueba.cargarNumeros()) {

if (prueba.isPrime(i)) {

System.out.println("El numero " + i + " es Primo");

} else {

System.out.print("El numero " + i + " no es Primo");

System.out.println(", la suma de sus digitos es : " + prueba.sumaDigitos(i));

}

}

}

}

**Class Agenda.java**

import java.util.ArrayList;

import javax.swing.JOptionPane;

public class Agenda implements java.io.Serializable {

final static String RUTA = "agenda.dat";

ArrayList<Persona> contactos;

public Agenda() {

contactos = new ArrayList();

}

public void verContactos() {

if (contactos.isEmpty()) {

System.out.println("Lista de Contactos Vacia.");

} else {

for (Persona a : contactos) {

System.out.println(a.tostring());

}

}

}

public ArrayList<Persona> Contactos() {

return contactos;

}

public void grabarAgenda() {

Archivo.guardar(RUTA, this);

}

void agregarContacto(Persona nuevo) {

contactos.add(nuevo);

}

}

class TestAgenda {

static Agenda prueba = new Agenda();

public static void main(String[] args) {

menu();

}

public static void menu() {

int opc;

do {

opc=0;

try {

opc = Integer.valueOf(JOptionPane.showInputDialog("1.Añadir Contacto\n"

+ "2.Ver Contactos\n"

+ "0.Salir"));

} catch (NumberFormatException e) {

}

switch (opc) {

case 1:

agregar();

break;

case 2:

prueba=(Agenda)Archivo.leer(Agenda.RUTA, prueba);

prueba.verContactos();

break;

case 0:

break;

default:

System.out.println("Opcion no Valida");

}

} while (opc != 0);

}

public static void agregar(){

Persona nuevo=new Persona();

nuevo.setNombre(JOptionPane.showInputDialog("Nombre"));

nuevo.setTelefono(JOptionPane.showInputDialog("Numero"));

nuevo.setDNI(JOptionPane.showInputDialog("DNI"));

nuevo.setDireccion(JOptionPane.showInputDialog("Direccion"));

prueba.agregarContacto(nuevo);

prueba.grabarAgenda();

}

}

**Class Persona.java**

public class Persona implements java.io.Serializable {

private String nombre;

private String DNI;

private String Direccion;

private String telefono;

public Persona(){

}

public Persona(String nombre, String telefono) {

this.nombre = nombre;

this.telefono = telefono;

}

public String tostring() {

return nombre + " " + telefono+ " " + DNI+ " " + Direccion;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public String getTelefono() {

return telefono;

}

public void setTelefono(String telefono) {

this.telefono = telefono;

}

public String getDNI() {

return DNI;

}

public void setDNI(String DNI) {

this.DNI = DNI;

}

public String getDireccion() {

return Direccion;

}

public void setDireccion(String Direccion) {

this.Direccion = Direccion;

}

}

**Class Prob6.java**

import java.util.ArrayList;

import javax.swing.\*;

class Prob6 {

double pagoHora = 5;

double PagoHoraExtra = pagoHora \* 1.5;

double horasTrabajadas = 85;

String RUTA = "Empleados.dat";

ArrayList<Empleado> Empleados;

double sueldoBruto(Empleado e) {

return e.getSueldoBase() + e.getHorasExtras() \* PagoHoraExtra;

}

public Prob6() {

Empleados = new ArrayList();

}

void grabarArchivo() {

Archivo.guardar(RUTA, Empleados);

}

void agregarEmpleado(Empleado a) {

for (Empleado e : Empleados) {

if (!e.getCodigo().equals(a.getCodigo())) {

Empleados.add(a);

}

else{

System.out.println("Codigo de Empleado ya existe.");

}

}

}

void abrirArchivo() {

Archivo.leer(RUTA, Empleados);

}

void calculoSueldoBruto() {

for (Empleado a : Empleados) {

System.out.println("El sueldo bruto de " + a.getNombre() + " es " + sueldoBruto(a));

}

}

double calculoPagoHora(Empleado e) {

return pagoHora = e.getSueldoBase() / horasTrabajadas;

}

void mostrarReporte() {

Empleado masGano = new Empleado();

Empleado menosGano;

try {

System.out.println("Reporte Empleados");

menosGano = Empleados.get(0);

for (Empleado e : Empleados) {

System.out.println(e);

if (sueldoBruto(e) > sueldoBruto(masGano)) {

masGano = e;

}

if (sueldoBruto(e) < sueldoBruto(menosGano)) {

menosGano = e;

}

}

System.out.println("El empleado que mas gano es " + masGano.getNombre() + " con un sueldo de " + sueldoBruto(masGano));

System.out.println("El empleado que menos gano es " + menosGano.getNombre() + " con un sueldo de " + sueldoBruto(menosGano));

} catch (NullPointerException e) {

System.out.println("Lista de Empleados vacia");

}

}

}

class TestProb6 {

static Prob6 test = new Prob6();

public static void main(String[] args) {

test.abrirArchivo();

menu();

}

public static void menu() {

int opc;

do {

opc = 0;

try {

opc = Integer.valueOf(JOptionPane.showInputDialog("1.Añadir Empleado\n"

+ "2.Mostrar Reporte\n"

+ "0.Salir"));

} catch (NumberFormatException e) {

}

switch (opc) {

case 1:

agregar();

break;

case 2:

test.mostrarReporte();

break;

case 0:

break;

default:

System.out.println("Opcion no Valida");

}

} while (opc != 0);

}

public static void agregar() {

Empleado nuevo = new Empleado();

nuevo.setNombre(JOptionPane.showInputDialog("Nombre"));

nuevo.setCodigo(JOptionPane.showInputDialog("Codigo"));

nuevo.setSueldoBase(Double.valueOf(JOptionPane.showInputDialog("SueldoBase")));

nuevo.setHorasExtras(Double.valueOf(JOptionPane.showInputDialog("HorasExtras")));

test.agregarEmpleado(nuevo);

test.grabarArchivo();

}

}

**Class Empleado.java**

import java.io.Serializable;

class Empleado implements Serializable {

private String codigo;

private String nombre;

private double SueldoBase;

private double horasExtras;

public Empleado(String codigo, String nombre, double SueldoBase, double horasExtras) {

this.codigo = codigo;

this.nombre = nombre;

this.SueldoBase = SueldoBase;

this.horasExtras = horasExtras;

}

public Empleado() {

this.codigo="Sin codigo";

this.nombre="Sin nombre";

this.SueldoBase=0;

this.horasExtras=0;

}

public double getHorasExtras() {

return horasExtras;

}

public void setHorasExtras(double horasExtras) {

this.horasExtras = horasExtras;

}

public String getCodigo() {

return codigo;

}

public void setCodigo(String codigo) {

this.codigo = codigo;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public double getSueldoBase() {

return SueldoBase;

}

public void setSueldoBase(double SueldoBase) {

this.SueldoBase = SueldoBase;

}

@Override

public String toString() {

return "Codigo = " + codigo + " nombre=" + nombre + " SueldoBase=" + SueldoBase + " horasExtras=" + horasExtras;

}

}