UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERIA

ESCUELA DE CIENCIAS Y SISTEMAS

LABORATORIO DE IPC 1

PRACTICA 2

MANUAL TECNICO

DENNIS MAURICIO CORADO MUÑÓZ

202010406

303232978108

INTRODUCCION

Este manual espera explicar de manera amplia y apropiada los métodos utilizados al momento de crear el programa de “Practica 2” para poder esclarecer dudas si se necesitara cambiar algo del respectivo código.

Se creo un archivo para la respectiva practica y se utilizo el paradigma de programación orientada a objetos para poder llevar a cabo el proyecto de mejor manera.

El proyecto se dividio en las siguientes clases:

Cada una de estas lleva información respectiva a los objetos que almacenaba y en la clase “ObjetosPoke” se controloba lo que era el llenar los arreglos que contendrían los respectivos objetos de cada clase y en la clase “Menu” se realizaban todas las operaciones para leer los archivos ingresados por el usuario.

De esta manera se llama a la clase menú para que el programa arranque.

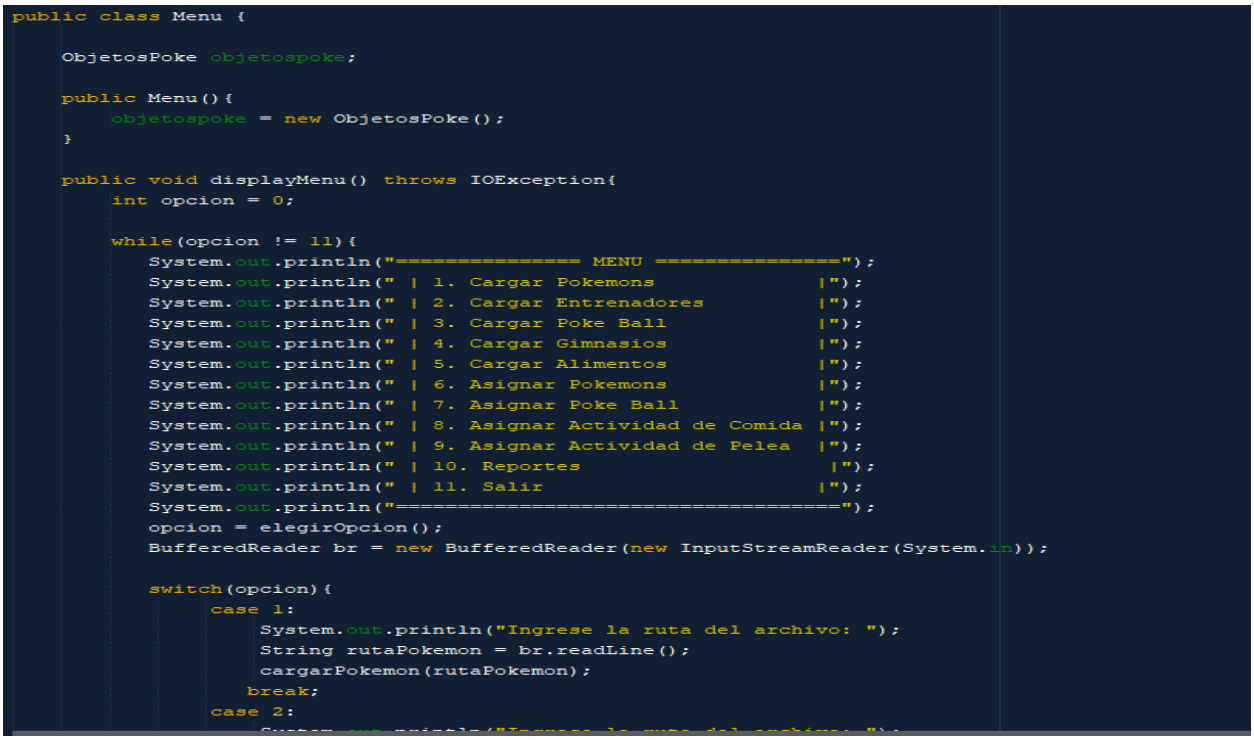
```
 */
package practica2;

import java.io.FileNotFoundException;
import java.io.IOException;

/**
 *
 * @author dennis
 */
public class Practica2 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) throws FileNotFoundException, IOException {

        Menu menu = new Menu();
        menu.displayMenu();

    }

}
```

Y en menú se encuentra lo siguiente:

Importación de las clases utilizadas

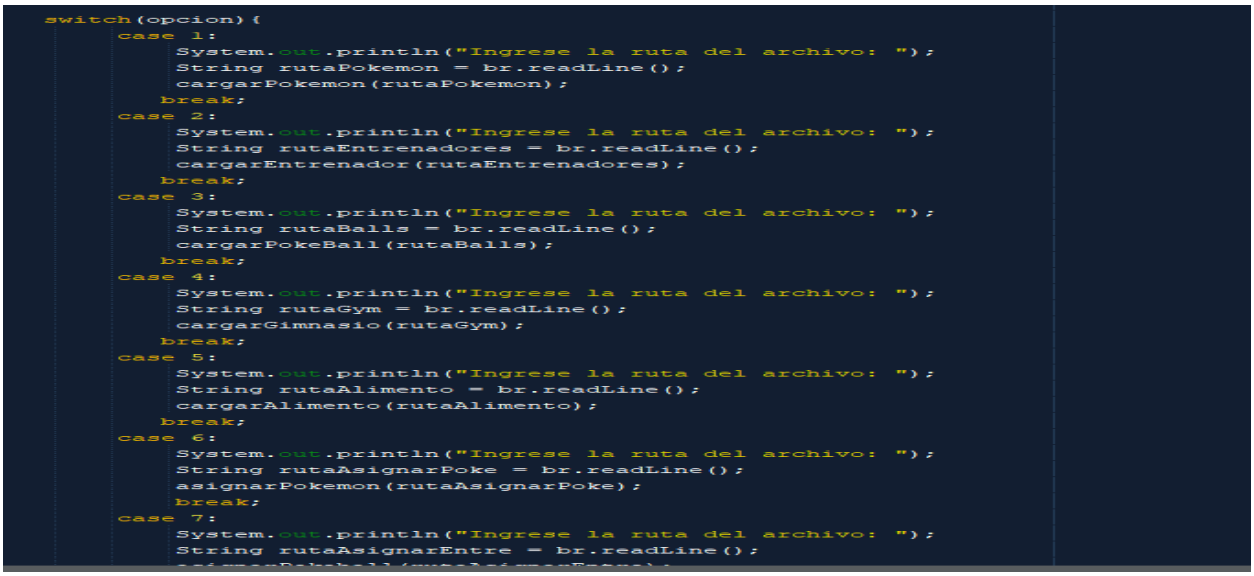
```
 */
package practica2;

import java.awt.Desktop;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.ObjectOutputStream;
import java.io.PrintWriter;

import java.util.Scanner;

/**
```

Inicialización de la clase “ObjetosPoke” para que esta entre en ejecución y pueda ser llamada en cada método así como creación de un switch para poder manejar las entradas del usuario



```
public class Menu {

    ObjetosPoke objetospoke;

    public Menu(){
        objetospoke = new ObjetosPoke();
    }

    public void displayMenu() throws IOException{
        int opcion = 0;

        while(opcion != 11){
            System.out.println("================ MENU ================");
            System.out.println(" | 1. Cargar Pokemons               |");
            System.out.println(" | 2. Cargar Entrenadores           |");
            System.out.println(" | 3. Cargar Poke Ball              |");
            System.out.println(" | 4. Cargar Gimnasios              |");
            System.out.println(" | 5. Cargar Alimentos              |");
            System.out.println(" | 6. Asignar Pokemons              |");
            System.out.println(" | 7. Asignar Poke Ball             |");
            System.out.println(" | 8. Asignar Actividad de Comida |");
            System.out.println(" | 9. Asignar Actividad de Pelea  |");
            System.out.println(" | 10. Reportes                      |");
            System.out.println(" | 11. Salir                        |");
            System.out.println("=====================================");
            opcion = elegirOpcion();
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

            switch(opcion){
                case 1:
                    System.out.println("Ingrese la ruta del archivo: ");
                    String rutaPokemon = br.readLine();
                    cargarPokemon(rutaPokemon);
                   break;
                case 2:
```

En el switch se llaman a las funciones que contienen cada proceso



```
    switch(opcion){
        case 1:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaPokemon = br.readLine();
            cargarPokemon(rutaPokemon);
           break;
        case 2:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaEntrenadores = br.readLine();
            cargarEntrenador(rutaEntrenadores);
           break;
        case 3:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaBalls = br.readLine();
            cargarPokeBall(rutaBalls);
           break;
        case 4:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaGym = br.readLine();
            cargarGimnasio(rutaGym);
           break;
        case 5:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaAlimento = br.readLine();
            cargarAlimento(rutaAlimento);
           break;
        case 6:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaAsignarPoke = br.readLine();
            asignarPokemon(rutaAsignarPoke);
            break;
        case 7:
            System.out.println("Ingrese la ruta del archivo: ");
            String rutaAsignarEntre = br.readLine();
```

Y tenemos declaraciones de las funciones que servirán para poder registrar la entrada del usuario a su vez como llenar los respectivos objetos su información

```
public int elegirOpcion(){
    Scanner sc = new Scanner(System.in);
    System.out.println("Seleccione la opcion a realizar: ");
    int op = Integer.parseInt(sc.nextLine());
    while(op < 1 || op > 10){
        System.out.println("Ingrese una opcion valida");
        op = Integer.parseInt(sc.nextLine());
    }
    return op;
}

public void cargarPokemon(String path) throws FileNotFoundException, IOException{
    Scanner scan = new Scanner(new BufferedReader(new FileReader(path)));

    String lineaEntrada = " ";
    String primerLinea = scan.nextLine();
    while(scan.hasNextLine()){
        lineaEntrada = scan.nextLine();
        if(!primerLinea.equals(lineaEntrada)){
            String []entrada = lineaEntrada.split(",");
            objetospoke.guardarPokemon(Integer.parseInt(entrada[0]), entrada[1], entrada[2], Double.parseDouble
        }
    }
    System.out.println("Pokemones Cargados");
}

public void cargarEntrenador(String path) throws FileNotFoundException{
    Scanner scan = new Scanner(new BufferedReader(new FileReader(path)));

    String lineaEntrada = " ";
    String primerLinea = scan.nextLine();
    while(scan.hasNextLine()){
        lineaEntrada = scan.nextLine();
        if(!primerLinea.equals(lineaEntrada)){
            String []entrada = lineaEntrada.split(",");
```

Los métodos de carga son todos muy similares a excepción de los parámetros que recibe cada objeto.

Para poder asignar objetos con otra información se utilizo funciones para permitir esta subida

```
public void asignarPokemon(String path) throws FileNotFoundException{
    Scanner scan = new Scanner(new BufferedReader(new FileReader(path)));
    String lineaEntrada = " ";
    String primerLinea = scan.nextLine();
    while(scan.hasNextLine()){
        lineaEntrada = scan.nextLine();
        if(!primerLinea.equals(lineaEntrada)){
            String []entrada = lineaEntrada.split(",");
            objetospoke.asignarPokemon(Integer.parseInt(entrada[0]), Integer.parseInt(entrada[1]));
        }
    }
    System.out.println("Pokemons asignados a su Pokeball");
}

public void asignarPokeball(String path) throws FileNotFoundException{
    Scanner scan = new Scanner(new BufferedReader(new FileReader(path)));
    String lineaEntrada = " ";
    String primerLinea = scan.nextLine();
    while(scan.hasNextLine()){
        lineaEntrada = scan.nextLine();
        if(!primerLinea.equals(lineaEntrada)){
            String []entrada = lineaEntrada.split(",");
            objetospoke.asignarPokeBall(Integer.parseInt(entrada[0]), Integer.parseInt(entrada[1]));
        }
    }
    System.out.println("Pokemons asignados a su Entrenador");
}

public void asignarAlimento(String path) throws FileNotFoundException{
    Scanner scan = new Scanner(new BufferedReader(new FileReader(path)));
    String lineaEntrada = " ";
    String primerLinea = scan.nextLine();
    while(scan.hasNextLine()){
        lineaEntrada = scan.nextLine();
        if(!primerLinea.equals(lineaEntrada)){
            String []entrada = lineaEntrada.split(",");
            objetospoke.asignarComida(Integer.parseInt(entrada[0]), Integer.parseInt(entrada[1]));
```

Y para la elaboración de reportes se creó su respectiva función haciendo uso de métodos que permiten la creación de archivos y escritura de estos y también se evidencia la creación de una función “Serializar” que creara un archivo binario que funcionara como un backup al momento de terminar la ejecución

```
public void serializar() throws FileNotFoundException, IOException{
    ObjectOutputStream os = new ObjectOutputStream(new FileOutputStream("objetosPoke.bin"));
    os.writeObject(objetospoke);
    os.close();
}
public void reportes() throws IOException{
    File file = new File("Reporte.html");

    if(!file.exists()){
        file.createNewFile();
    }
    PrintWriter pw = new PrintWriter(file);
    char comillas = '"';
    pw.println("<style>"
            + "body{"
            + "background: url(fondo.gif);"
            + "font-family: Arial;"
            + "background-size: 1600px 800px;"
            + "}"
            + "h1{"
            + "text-align: center;"
            + "}"
            + "#container{"
            + "background: white;"
            + "opacity: 0.8;"
            + "}"
            + "h2{"
            + "text-align: center;"
            + "background: orange;"
            + "}"
            + "</style>");
    pw.println("<div id= " + comillas + "container" + comillas + ">");
    pw.println("<h1>Reportes Pokemon</h1>");
    pw.println("<h2>Reporte de Entrenadores</h2>");
    pw.println("<div>");
    for (int i = 0; i < objetospoke.contadorEntrenadores; i++) {
        if(objetospoke.entrenadores[i].getPokeball() != null){
```

En la clase “ObjetosPoke” se encuentran los arreglos y métodos que funcionar para cargar los arreglos así como asignar a los objetos una propiedad extra

```
import java.io.Serializable;
/**
 *
 * @author dennis
 */
public class ObjetosPoke implements Serializable{

    int contadorPokemon;
    int contadorPokeBall;
    int contadorAlimentos;
    int contadorEntrenadores;
    int contadorGimnasios;
    public Pokemon [] pokemon = new Pokemon[150];
    public Entrenadores [] entrenadores = new Entrenadores[25];
    public PokeBall [] pokeball = new PokeBall[150];
    public Gimnasios [] gimnasios = new Gimnasios[25];
    public Alimentos [] alimentos = new Alimentos[15];

    public ObjetosPoke(){
        contadorPokemon = 0;
        contadorPokeBall = 0;
        contadorAlimentos = 0;
        contadorEntrenadores = 0;
        contadorGimnasios = 0;
    }

    public void guardarPokemon(int id, String tipo, String nombre, double vida, double ptsAtq, boolean capturado, boolean estado){
        Pokemon nuevoPokemon = new Pokemon(id, tipo, nombre, vida, ptsAtq, capturado, estado);
        pokemon[contadorPokemon] = nuevoPokemon;
        contadorPokemon++;
    }
```

Metodo para utilizado para asignar atributo extra a cada objetos respectivo esta forma de asignar atributos extras es la misma utilizada en cada objeto diferente

```
public void asignarPokeBall(int idEntrenador, int idBall){
    Entrenadores entrenador_asignable = getEntrenador(idEntrenador);
    PokeBall pokeball_asignable = getPokeball(idBall);
    entrenador_asignable.setPokeball(pokeball_asignable);
}

public Entrenadores getEntrenador(int id){
    for (int i = 0; i < entrenadores.length; i++) {
        if (id == entrenadores[i].getId()) {
            return entrenadores[i];
        }
    }
    return null;
}

public void asignarComida(int idAct, int idPoke){
    Alimentos alimento_asignable = getAlimento(idAct);
    Pokemon pokemon_asignable = getPokemonAl(idPoke);
    pokemon_asignable.setAlimento(alimento_asignable);
}

public Alimentos getAlimento(int id){
    for (int i = 0; i < alimentos.length; i++) {
        if(id == alimentos[i].getId()){
            return alimentos[i];
        }
    }
    return null;
}
```

Posteriormente se tienen las clases que contienen sus objetos con sus respectivos Getter y setters para poder almacenar la información asi como alguno de estos tienen un objeto extra de una clase diferente que será utilizado cuando se suba la información.

```
public class Pokemon implements Serializable{

    int id;
    String tipo;
    String nombre;
    double vida;
    double ptsAtq;
    boolean capturado;
    boolean estado;
    Alimentos alimento;

    public Pokemon(int id, String tipo, String nombre, double vida, double ptsAtq, boolean capturado, boolean estado) {
        this.id = id;
        this.tipo = tipo;
        this.nombre = nombre;
        this.vida = vida;
        this.ptsAtq = ptsAtq;
        this.capturado = capturado;
        this.estado = estado;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getTipo() {
        return tipo;
```

```java
public boolean isCapturado() {
    return capturado;
}

public void setCapturado(boolean capturado) {
    this.capturado = capturado;
}

public boolean isEstado() {
    return estado;
}

public void setEstado(boolean estado) {
    this.estado = estado;
}

public Alimentos getAlimento() {
    return alimento;
}

public void setAlimento(Alimentos alimento) {
    this.alimento = alimento;
}
```