Manual técnico

Piscifactoría

Lois Domínguez Domínguez, Manuel Bértolo Blanco, Miriam Betanzos Jamardo

Componentes

Sistema de monedas

Componente que proporciona un sistema que gestiona las monedas de las que dispone el usuario.

Clases

SistemaMonedas

Clase pública que representa a un sistema en el que se dispone de una cierta cantidad de monedas que se puede establecer y obtener.

Atributos

private int monedas

Cantidad de monedas de las que se dispone.

Métodos

SistemaMonedas()

public SistemaMonedas()
Constructor de un objeto de la clase con una cantidad nula de monedas.

SistemaMonedas(int monedas)

public SistemaMonedas(int monedas)
Constructor de un objeto de la clase con una cantidad determinada de monedas.

Parámetros

monedas Cantidad de monedas con las que se inicializa el sistema de monedas.

getMonedas()

public int getMonedas()
Getter del atributo monedas.

devuelve Cantidad de monedas disponible en el sistema de monedas.

setMonedas(int monedas)

public void setMonedas(int monedas)
Setter del atributo monedas.

Parámetros

monedas Cantidad de monedas disponibles a establecer en el sistema de monedas.

toString()

public String toString()
Devuelve un string con información relevante del sistema de monedas. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante del sistema de monedas.

Clases

AdaptadorJson

Clase interna privada que implementa las interfaces JsonSerializer<SistemaMonedas> y JsonDeserializer<SistemaMonedas>. Se encarga de adaptar la serialización y deserialización de la clase SistemaMonedas al formato JSON.

Métodos

deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context)

public SistemaMonedas deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context) throws JsonParseException
Se encarga de la deserialización de un objeto SistemaMonedas.

Parámetros

json Elemento json que contiene la información del objeto SistemaMonedas.
typeOfT Tipo del elemento a deserializar.
context Contexto de deserialización

devuelve Objeto SistemaMonedas deserializado.

lanza JsonParseException Cuando no se puede deserializar el objeto.

serialize(SistemaMonedas src, Type typeOfSrc, JsonSerializationContext context)

public JsonElement serialize(SistemaMonedas src, Type typeOfSrc, JsonSerializationContext context)
Se encarga de la serialización de un objeto SistemaMonedas.

Parámetros

src Objeto SistemaMonedas a serializar.
typeOfSrc Tipo del objeto a serializar.
context Contexto de serialización.

devuelve Elemento json con toda la información del objeto SistemaMonedas serializada.

Generador de menús

Componente encargado de proporcionar herramientas para generar menús, este depende del componente del sistema de entrada.

Clases

GeneradorMenus

Clase pública que proporciona una serie de utilidades para la generación de menús.

Métodos

generarMenu(String[] opciones, int numeroOpcionInicial)

public static void generarMenu(String[] opciones, int numeroOpcionInicial)
Genera un menú con unas opciones.

Parámetros

opciones Opciones del menú a generar.
numeroOpcionInicial Número de la opción inicial del menú.

generarMenu(String[] cabecera, String[] opciones, int numeroOpcionInicial)

public static void generarMenu(String[] cabecera, String[] opciones, int numeroOpcionInicial)
Genera un menú con una cabecera y unas opciones.

Parámetros

cabecera Cabecera del menú a generar.
opciones Opciones del menú a generar.
numeroOpcionInicial Número de la opción inicial del menú.

generarMenuOperativo(String[] opciones, int numeroOpcionInicial, int numeroOpcionFinal)

public static int generarMenuOperativo(String[] opciones, int numeroOpcionInicial, int numeroOpcionFinal)
Genera un menú operativo con unas determinadas opciones.

Parámetros

opciones Opciones del menú a generar.
numeroOpcionInicial Número de la opción inicial del menú.
numeroOpcionFinal Número de la opción final del menú.

devuelve Opción seleccionada por el usuario.

generarMenuOperativo(String[] cabecera, String[] opciones, int numeroOpcionInicial, int numeroOpcionFinal)

public static int generarMenuOperativo(String[] cabecera, String[] opciones, int numeroOpcionInicial, int numeroOpcionFinal)
Genera un menú operativo con una cabecera y unas opciones.

Parámetros

cabecera Cabecera del menú a generar.
opciones Opciones del menú a generar.
numeroOpcionInicial Número de la opción inicial del menú.
numeroOpcionFinal Número de la opción final del menú.

devuelve Opción seleccionada por el usuario.

toString()

public String toString()
Devuelve un string con información relevante de la clase. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante de la clase.

Sistema de entrada

Componente que se encarga de la gestión de la entrada de datos. Este es utilizado por el componente de generador de menús.

Clases

SistemaEntrada

Clase pública que se encarga de gestionar la entrada de datos introducidos por el usuario.

Atributos

private static final BufferedReader BUFER_ENTRADA

Búfer de entrada de datos introducidos por el usuario por teclado.

Métodos

entradaOpcionNumerica(int minimo, int maximo)

public static int entradaOpcionNumerica(int minimo, int maximo)
Gestiona la entrada de opciones numéricas siendo aceptables un cierto rango de opciones.

Parámetros

minimo Valor mínimo aceptado.
maximo Valor máximo aceptado.

devuelve Opción escogida por el usuario.

entradaOpcionNumerica(int[] opciones)

public static int entradaOpcionNumerica(int[] opciones)
Gestiona la entrada de opciones numéricas.

Parámetros

opciones Opciones numéricas aceptadas.

devuelve Opción numérica aceptada escogida por el usuario.

entradaTexto()

public static String entradaTexto()
Gestiona la entrada de texto introducido por el usuario.

devuelve Texto introducido por el usuario.

entradaOpcionNumericaEnteraPositiva()

public static int entradaOpcionNumericaEnteraPositiva()
Gestiona la entrada de un número entero positivo introducido por el usuario.

> **devuelve** Número entero positivo introducido por el usuario.

close()

public static void close()
Cierra el búfer de entrada de datos.

toString()

public String toString()
Devuelve un string con información relevante de la clase. Este método sobrescribe al método de la superclase Object.

> **devuelve** String con información relevante de la clase.

FechaTiempoLocal

Componente que contiene utilidades relativas a la fecha y a la hora.

Clases

FechaTiempoLocal

Clase pública que contiene utilidades relativas a la fecha y a la hora.

Métodos

obtenerFechaTiempoActual()

public static String obtenerFechaTiempoActual()
Permite obtener la fecha y tiempo del sistema en formato [aaaa-mm-dd hh:mm:ss].

> **devuelve** Fecha y tiempo del sistema en formato [aaaa-mm-dd hh:mm:ss].

SistemaFicheros

Componente que se encarga de la gestión del sistema de ficheros.

Clases

SistemaFicheros

Clase pública con utilidades relativas al sistema de ficheros.

Métodos

crearArchivo(String rutaArchivo)

public static void crearArchivo(String rutaArchivo) throws IOException
Crea un archivo en el sistema de ficheros a partir de su ruta.

Parámetros

rutaArchivo Ruta del archivo a crear.

Lanza IOException Cuando no se ha podido crear el archivo en el sistema de ficheros.

crearCarpeta(String rutaCarpeta)

public static void crearCarpeta(String rutaCarpeta) throws IOException
Crea una carpeta en el sistema de ficheros a partir de su ruta.

Parámetros

rutaCarpeta Ruta de la carpeta a crear.

Lanza IOException Cuando no se ha podido crear la carpeta en el sistema de ficheros.

crearEstructuraCarpetas(String estructuraCarpetas)

public static void crearEstructuraCarpetas(String estructuraCarpetas) throws IOException
Crea una estructura de carpetas en el sistema de ficheros a partir de la ruta de la estructura de carpetas.

Parámetros

estructuraCarpetas Ruta de las estructura de carpetas.

Lanza IOException Cuando no se ha podido crear la estructura de carpetas en el sistema de ficheros.

existeArchivo(String rutaArchivo)

public static boolean existeArchivo(String rutaArchivo) throws IOException
Permite comprobar la existencia de un archivo en el sistema de ficheros a partir de su ruta.

Parámetros

rutaArchivo Ruta del archivo a comprobar su existencia.

devuelve Si el archivo existe o no.

Lanza IOException Si el archivo del que se comprueba su existencia se trata de un directorio.

existeDirectorio(String rutaDirectorio)

public static boolean existeDirectorio(String rutaDirectorio) throws IOException
Permite comprobar la existencia de un directorio en el sistema de ficheros a partir de su ruta.

Parámetros

rutaDirectorio Ruta del directorio a comprobar su existencia.

devuelve Si el directorio existe o no.

Lanza IOException Si el directorio del que se comprueba su existencia se trata de un archivo.

isDirectorioVacio(String rutaDirectorio)

public static boolean isDirectorioVacio(String rutaDirectorio) throws IOException
Indica si un directorio del sistema de ficheros está vacío.

Parámetros

rutaDirectorio Ruta del directorio a comprobar si está vacío.

devuelve Si el directorio está vacío o no.

Lanza IOException Si el directorio no existe o si se trata de un fichero.

borrarArchivo(String rutaArchivo)

public static void borrarArchivo(String rutaArchivo) throws IOException
Permite borrar un archivo del sistema de ficheros.

Parámetros

rutaArchivo Ruta del archivo a borrar.

Lanza IOException Cuando el archivo no ha podido ser eliminado o no existe.

LecturaEscrituraFicherosPlanos

Componentes con utilidades para la escritura y lectura de ficheros de texto plano.

Clases

LecturaEscrituraFicherosPlanos

Clase con utilidades para la lectura y escritura de ficheros de texto plano.

Métodos

escrituraFicheroTextoPlano(File archivo, String texto, String codificacion)

public static void escrituraFicheroTextoPlano(File archivo, String texto, String codificacion) throws IOException
Permite la escritura de un texto en un archivo de texto plano.

Parámetros

archivo Archivo en el que se realizará la escritura del texto.
texto Texto que se escribirá en el archivo.
codificacion Codificación en la que se encuentra el texto a escribir en el fichero.

Lanza IOException Cuando surge un error al escribir en el archivo o cuando surge un error al cerrar el búfer de escritura.

escrituraFicheroTextoPlanoSinSobreescritura(File archivo, String texto, String codificacion)

public static void escrituraFicheroTextoPlanoSinSobreescritura(File archivo, String texto, String codificacion) throws IOException
Permite la escritura de un texto en un archivo de texto plano sin eliminar el contenido escrito anteriormente.

Parámetros

archivo Archivo en el que se realizará la escritura del texto.
texto Texto que se escribirá en el archivo.
codificacion Codificación en la que se encuentra el texto a escribir en el fichero.

Lanza IOException Cuando surge un error al escribir en el archivo o cuando surge un error al cerrar el búfer de escritura.

lecturaFicheroTextoPlano(File archivo, String codificacion)

public static String lecturaFicheroTextoPlano(File archivo, String codificacion) throws IOException
Permite la lectura de un fichero de texto plano.

Parámetros

archivo Archivo de texto plano del cuál se lee el texto.
codificacion Codificación del texto del archivo.

devuelve Texto que contiene el archivo de texto plano.

Lanza IOException Cuando surge un error al leer el texto o al cerrar el búfer de lectura.

LecturaEscrituraJSON

Componente encargado de la lectura y escritura de archivos JSON.

Clases

LecturaEscrituraJSON

Clase para gestionar la lectura y escritura de ficheros JSON.

Métodos

guardarJSON(File archivo, T objeto)

public static <T> void guardarJSON(File archivo, T objeto) throws IOException
Guarda un objeto en un archivo JSON.

Parámetros

archivo Archivo donde se guardará el objeto JSON.
objeto Objeto a guardar.

Lanza IOException Si ocurre un error al guardar el archivo.

cargarJSON(File archivo)

public static void <T> T cargarJSON(File archivo) throws IOException
Carga un objeto desde un archivo JSON.

Parámetros

archivo Archivo JSON desde donde se cargará el objeto.
<T> Tipo del objeto a cargar.
Class<T> Clase del objeto que se quiere cargar.

devuelve El objeto cargado.

lanza IOException Si ocurre un error al leer el archivo.

Transcripciones

Componente que se encarga de realizar las transcripciones del simulador.

Clases

Transcripciones

Clase con utilidades para las transcripciones de una partida del sistema.

Atributos

private File archivoTranscripciones

Archivo del sistema de archivos donde se realizarán las transcripciones del sistema.

Métodos

Transcripciones(File archivoTranscripciones)

Constructor de objetos de la clase.

Parámetros:

> **archivoTranscripciones** Archivo donde se realizarán las transcripciones del sistema.

iniciarTranscripciones(String nombrePartida, int dinero, String[] pecesRio, String[] pecesMar, String piscifactoriaInicial)

public void iniciarTranscripciones(String nombrePartida, int dinero, String[] pecesRio, String[] pecesMar, String piscifactoriaInicial)
Inicia el informe con información inicial de la partida.

Parámetros:

> **nombrePartida** Nombre de la partida de la que se realizan las transcripciones.
> **dinero** Dinero del que se dispone inicialmente.
> **pecesRio** Peces de río implementados en la simulación.
> **pecesMar** Peces de mar implementados en la simulación.
> **piscifactoriaInicial** Piscifactoría inicial con la que se inicia la simulación.

iniciarTranscripciones(String nombrePartida, int dinero, String[] pecesRio, String[] pecesMar, String[] extras, String piscifactoriaInicial)

public void iniciarTranscripciones(String nombrePartida, int dinero, String[] pecesRio, String[] pecesMar, String[] extras, String piscifactoriaInicial)
Inicia el informe con información inicial de la partida.

Parámetros:

nombrePartida Nombre de la partida de la que se realizan las transcripciones.
dinero Dinero del que se dispone inicialmente.
pecesRio Peces de río implementados en la simulación.
pecesMar Peces de mar implementados en la simulación.
extras Extras implementados en la simulación.
piscifactoriaInicial Piscifactoría inicial con la que se inicia la simulación.

registrarCompraComida(int cantidadComida, String tipoComida, int monedas, String piscifactoria)

public void registrarCompraComida(int cantidadComida, String tipoComida, int monedas, String piscifactoria)
Registra la compra de una cantidad de comida para una piscifactoría.

Parámetros:

cantidadComida Cantidad de comida comprada.
tipoComida Tipo de comida comprada.
monedas Monedas por las que se compró la comida.
piscifactoria Piscifactoría a la que se le compró la comida.

registrarCompraComida(int cantidadComida, String tipoComida, int monedas)

public void registrarCompraComida(int cantidadComida, String tipoComida, int monedas)
Registra la compra de una cantidad de comida para el almacén central.

Parámetros:

cantidadComida Cantidad de comida comprada.
tipoComida Tipo de comida comprada.
monedas Monedas por las que se compró la comida.

registrarCompraPeces(String pez, boolean sexo, int monedas, int tanque, String piscifactoria)

public void registrarCompraPeces(String pez, boolean sexo, int monedas, int tanque, String piscifactoria)
Registra la compra de un pez en el archivo de transcripciones.

Parámetros:

pez Especie del pez comprado.
sexo Sexo del pez comprado.
tanque Tanque donde el pez fue incorporado.
piscifactoria Piscifactoría donde el pez fue incorporado.

registrarVentaPeces(int pecesVendidos, String piscifactoria, int monedas)

public void registrarVentaPeces(int pecesVendidos, String piscifactoria, int monedas)
Registra una venta de peces en el archivo de transcripciones.

Parámetros:

pecesVendidos Número de peces vendidos.
piscifactoria Piscifactoría de la que se venden los peces.
monedas Monedas obtenidas con la venta.

registrarLimpiezaTanque(int tanque, String piscifactoria)

public void registrarLimpiezaTanque(int tanque, String piscifactoria)
Registra la limpieza de peces de un tanque.

Parámetros:

tanque Tanque que se ha limpiado.
piscifactoria Piscifactoría a la que pertenece el tanque que se ha limpiado.

registrarVaciadoTanque(int tanque, String piscifactoria)

public void registrarVaciadoTanque(int tanque, String piscifactoria)
Registra el vaciado de un tanque.

Parámetros:

tanque Tanque que se ha vaciado.
piscifactoria Piscifactoría a la que pertenece el tanque que se ha vaciado.

registrarCompraPiscifactoria(int tipo, String piscifactoria, int monedas)

public void registrarCompraPiscifactoria(int tipo, String piscifactoria, int monedas)
Registra la compra de una piscifactoría.

Parámetros:

tipo Tipo de la piscifactoría si es 0 es de río y si es 1 es de mar.
piscifactoria Nombre de la piscifactoría comprada.
monedas Monedas por las que se compró la piscifactoría.

resgistrarCompraTanque(int numeroTanque, String piscifactoria, int monedas)

public void registrarCompraTanque(int numeroTanque, String piscifactoria, int monedas)
Registra la compra de un tanque.

Parámetros:

numeroTanque Número del tanque comprado.
piscifactoria Piscifactoría para la que se compró el tanque.
monedas Monedas por las que se compró el tanque.

registrarCompraAlmacenCentral()

public void registrarCompraAlmacenCentral()
Registra la compra del almacén central.

registrarMejoraPiscifactoria(String piscifactoria, int nuevaCapacidad, int monedas)

public void registrarMejoraPiscifactoria(String piscifactoria, int nuevaCapacidad, int monedas)
Registra la mejora de una piscifactoría.

Parámetros:

piscifactoria Piscifactoría a la que se le realiza la mejora.
nuevaCapacidad Nueva capacidad de comida de la piscifactoría.
monedas Monedas que costó la realización de la mejora.

registrarMejoraAlmacenCentral(int nuevaCapacidad, int monedas)

public void registrarMejoraAlmacenCentral(int nuevaCapacidad, int monedas)
Registra la mejora del almacén central.

Parámetros:
nuevaCapacidad Nueva capacidad de comida del almacén central
monedas Monedas por las que se realizó la mejora.

registrarPasoDia(int dia, int pecesDeRio, int pecesDeMar, int monedas, int pecesVendidos)

public void registrarPasoDia(int dia, int pecesDeRio, int pecesDeMar, int monedas, int pecesVendidos)
Registra el paso de un día.

Parámetros:

dia Día que se acaba.
pecesDeRio Peces de río que hay en la simulación después de pasar el día.
pecesDeMar Peces de mar que hay en la simulación después de pasar el día.
monedas Monedas obtenidas por la venta de peces al pasar el día.
pecesVendidos Número de peces vendidos al pasar el día.

registrarAnadirPecesOculto(String piscifactoria)

public void registrarAnadirPecesOculto(String piscifactoria)
Registra el uso de la opción oculta para añadir peces.

Parámetros:

piscifactoria Piscifactoría a la que se añadieron los peces mediante la opción oculta.

registrarCreacionRecompensa(String recompensa)

public void registrarCreacionRecompensa(String recompensa)
Registra la creación de una recompensa.

Parámetros:

recompensa Recompensa creada.

registrarUsoRecompensa(String recompensa)

public void registrarUsoRecompensa(String recompensa)
Registra el uso de una recompensa.

Parámetros:

recompensa Recompensa usada.

registrarAnadirMonedasOculta(int monedasActuales)

public void registrarAnadirMonedasOculta(int monedasActuales)
Registra el añadido de monedas mediante la opción oculta.

Parámetros:

monedasActuales Monedas de las que se dispone tras el añadido de monedas.

Logs

Componente que se encarga del fichero de *logs* de una partida.

Clases

Logs

Atributos

private File archivoLogs

Archivo donde se escriben los *logs* de la partida.

Métodos

Logs(File archivoLogs)

public Logs(File archivoLogs)
Constructor de objetos de la clase parametrizado.

Parámetros

archivoLogs Archivo donde se escriben los *logs* de la partida.

inicioLog(String partida, String piscifactoria)

public void inicioLog(String partida, String piscifactoria)
Inicia el archivo de *logs* de una partida del sistema.

Parámetros

partida Nombre de la partida.
piscifactoria Nombre de la piscifactoría inicial de la partida.

registrarCompraComida(int cantidad, String tipo, String piscifactoria)

public void registrarCompraComida(int cantidad, String tipo, String piscifactoria)
Registra la compra de una cantidad de comida a una piscifactoría determinada.

Parámetros

cantidad Cantidad de comida adquirida.
tipo Tipo de comida adquirida.
piscifactoria Piscifactoría a la que se le compró la comida.

registrarCompraComida(int cantidad, String tipo)

public void registrarCompraComida(int cantidad, String tipo)
Registra la compra de una cantidad de comida para el almacén central.

Parámetros

cantidad Cantidad de comida adquirida.
tipo Tipo de comida adquirida.

registrarCompraPez(String pez, boolean sexo, int tanque, String piscifactoria)

public void registrarCompraPez(String pez, boolean sexo, int tanque, String piscifactoria)
Registra la compra de un pez.

Parámetros

pez Especie del pez que se ha comprado.
sexo Sexo del pez que se ha comprado. Si es *true* es hembra y si es *false* es macho.
tanque Número del tanque donde se incorpora el pez.
piscifactoria Nombre de la piscifactoría a la que se incorpora el pez.

registrarVentaPeces(int pecesVendidos, String piscifactoria)

public void registrarVentaPeces(int pecesVendidos, String piscifactoria)
Registra la venta manual de peces de una piscifactoría.

Parámetros

pecesVendidos Número de peces vendidos.
piscifactoria Nombre de la piscifactoría a la que pertenecían los peces.

registrarLimpiezaTanque(int tanque, String piscifactoria)

public void registrarLimpiezaTanque(int tanque, String piscifactoria)
Registra la limpieza realizada en un tanque.

Parámetros

tanque Número del tanque al que se le realizó la limpieza.
piscifactoria Nombre de la piscifactoría a la que pertenecía el tanque.

registrarVaciadoTanque(int tanque, String piscifactoria)

public void registrarVaciadoTanque(int tanque, String piscifactoria)
Registra el vaciado de un tanque.

Parámetros

tanque Número del tanque vaciado.
piscifactoria Nombre de la piscifactoría a la que pertenecía el tanque.

registrarCompraPiscifactoria(int tipo, String piscifactoria)

public void registrarCompraPiscifactoria(int tipo, String piscifactoria)
Registra la compra de una piscifactoría.

Parámetros

tipo Tipo de la piscifactoría si es 0 es de río y si es 1 es de mar.
piscifactoria Nombre de la piscifactoría comprada.

registrarCompraTanque(String piscifactoria)

public void registrarCompraTanque(String piscifactoria)
Registra la compra de un tanque para una piscifactoría.

Parámetros

piscifactoria Nombre de la piscifactoría a la que se le compró el tanque.

registrarCompraAlmacenCentral()

public void registrarCompraAlmacenCentral()
Registra la compra del almacén central.

registrarMejoraPiscifactoria(String piscifactoria)

public void registrarMejoraPiscifactoria(String piscifactoria)
Registra una mejora realizada a una piscifactoría.

Parámetros

piscifactoria Nombre de la piscifactoría a la que se le realizó la mejora.

registrarMejoraAlmacenCentral()

public void registrarMejoraAlmacenCentral()
Registra una mejora realizada al almacén central.

registrarPasoDia(int dia)

public void registrarPasoDia(int dia)
Registra el paso de un día.

Parámetros

dia Día que se acaba.

registrarAnadirPecesOculto(String piscifactoria)

public void registrarAnadirPecesOculto(String piscifactoria)
Registra el uso de la opción oculta para añadir peces a una determinada piscifactoría.

Parámetros

piscifactoria Nombre de la piscifactoría a la que se le añaden los peces mediante la opción oculta.

registrarAnadirMonedasOculto()

public void registrarAnadirMonedasOculto()
Registrar el uso de la opción oculta para obtener monedas.

registrarSalidaPartida()

public void registrarSalidaPartida()
Registra la salida de la partida.

registrarCreacionRecompensa()

public void registrarCreacionRecompensa()
Registra la creación de una recompensa.

registrarRecompensaRecibida(String receptor)

public void registrarRecompensaRecibida(String receptor)
Registra la recepción de una recompensa.

Parámetros
receptor Receptor de la recompensa.

registrarUsoRecompensa(String recompensa)

public void registrarUsoRecompensa(String recompensa)
Registra el uso de una recompensa.

Parámetros
recompensa Recompensa usada.

obtenerFechaHora()

public static String obtenerFechaHora()
Permite obtener la fecha y hora del sistema en formato [aaaa-mm-dd hh:mm:ss].

devuelve Fecha y hora del sistema en formato [aaaa-mm-dd hh:mm:ss].

Clases propias

Clase Pez

Clase abstracta que representa la base de un pez del sistema.

Atributos

protected final String nombre

Nombre común del pez.

protected final String nombreCientifico

Nombre científico del pez.

protected int edad

Edad del pez en días.

protected final boolean sexo

Sexo del pez. True indica que el pez es hembra y false indica que el pez es macho.

protected boolean fertil

Indica si el pez es fértil y puede reproducirse.

protected boolean vivo

Indica si el pez está vivo.

protected boolean alimentado

Indica si el pez ha sido alimentado.

protected int diasSinReproducirse

Número de días en los que el pez no se ha reproducido desde la última vez que era fértil.

Métodos

getNombre()

public String getNombre()
Getter del nombre común del pez.

> **devuelve** Nombre común del pez.

getNombreCientifico()

public String getNombreCientifico()
Getter del nombre científico del pez.

devuelve Nombre científico del pez.

getEdad()

public int getEdad()
Getter de la edad del pez en días.

devuelve Edad del pez en días.

isSexo()

public boolean isSexo()
Getter del sexo del pez.

devuelve True si el pez es hembra y false si el pez es macho.

isFertil()

public boolean isFertil()
Getter que indica si el pez es fértil y puede reproducirse.

devuelve True si el pez es fértil y puede reproducirse.

isVivo()

public boolean isVivo()
Getter que indica si el pez está vivo o no.

devuelve True si el pez está vivo.

isAlimentado()

public boolean isAlimentado()
Getter que indica si el pez ha sido alimentado o no.

devuelve True si el pez ha sido alimentado.

getDiasSinReproducirse()

public int getDiasSinReproducirse()
Getter del número de días en los que el pez no se ha reproducido desde la última vez que ha sido fértil.

devuelve Número de días en los que el pez no se ha reproducido desde la última vez que ha sido fértil.

setEdad(int edad)

public void setEdad(int edad)
Setter de la edad del pez en días.

Parámetros
edad Edad del pez en días a establecer.

setFertil(boolean fertil)

public void setFertil(boolean fertil)
Setter de si el pez es fértil y puede reproducirse.

Parámetros
fertil Si es true se establece que el pez es fértil y puede reproducirse.

setVivo(boolean vivo)

public void setVivo(boolean vivo)
Setter de si el pez está vivo.

Parámetros
vivo Si es true se establece que el pez está vivo.

setAlimentado(boolean alimentado)

public void setAlimentado(boolean alimentado)
Setter de si el pez ha sido alimentado.

Parámetros
alimentado Si es true se establece que el pez ha sido alimentado.

setDiasSinReproducirse(int diasSinReproducirse)

public void setDiasSinReproducirse(int diasSinReproducirse)
Setter del número de días sin reproducirse desde que el pez era fértil.

Parámetros
diasSinReproducirse Número de días sin reproducirse desde que el pez era fértil.

Pez(String nombre, String nombreCientifico, boolean sexo)

protected Pez(String nombre, String nombreCientifico, boolean sexo)
Constructor de peces a partir de su nombre, nombre científico y sexo.

Parámetros
nombre Nombre del pez.
nombreCientifico Nombre científico del pez.
sexo Sexo del pez si es true el pez es hembra y si es false el pez es macho.

showStatus()

public abstract void showStatus()
Muestra el estado del pez.

grow()

public abstract void grow()
Hace que el pez crezca un día.

reset()

public void reset()
Reinicia el pez estableciendo sus atributos a sus estados iniciales.

comer()

public int comer()
Indica la cantidad de comida que consume el pez un determinado día.

devuelve Cantidad de comida que consume el pez un determinado día.

isMaduro()

public abstract boolean isMaduro()
Indica si el pez está maduro.

devuelve True si el pez está maduro.

isEdadOptima()

public abstract boolean isEdadOptima()
Indica si el pez está en la edad óptima para ser vendido.

devuelve True si el pez está en la edad óptima para ser vendido.

obtenerPezHijo()

public abstract Pez obtenerPezHijo()
Devuelve un pez recién nacido de sexo masculino.

devuelve Pez recién nacido de sexo masculino.

obtenerPezHija()

public abstract Pez obtenerPezHija()
Devuelve un pez recién nacido de sexo femenino.

devuelve Pez recién nacido de sexo femenino.

toString()

public String toString()
Devuelve un string con información relevante del pez. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante del pez.

Clases

AdaptadorJSON

Clase privada que implementa las interfaces JsonSerializer<Pez> y JsonDeserializar<Pez> que se encarga de adaptar la serialización y deserialización de un objeto Pez al formato json.

Métodos

deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context)

public Pez deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context) throws JsonParseException
Se encarga de la deserialización de un objeto Pez.

Parámetros
json Elemento json que contiene la información del objeto Pez.
typeOfT Tipo del objeto a deserializar.
context Contexto de la serialización.

devuelve Objeto Pez serializado.

lanza JsonParseException Cuando no se puede deserializar el objeto Pez.

serialize(Pez src, Type typeOfSrc, JsonSerializationContext context)

public JsonElement serialize(Pez src, Type typeOfSrc, JsonSerializationContext context)

Parámetros
src Objeto Pez a serializar.
typeOfSrc Tipo del objeto a serializar.
context Contexto de la serialización,

devuelve Elemento json con la información del objeto Pez serializada.

Peces

Clase Carnívoro

Clase abstracta pública que extiende de Pez la cual representa un pez con tipo de alimentación carnívora.

Métodos

Carnivoro(String nombre, String nombreCientifico, boolean sexo)

protected Carnivoro(String nombre, String nombreCientifico, boolean sexo)
Constructor de la clase carnívoro el cual llama al constructor de la superclase.

Parámetros

nombre Nombre del pez.
nombreCientifico Nombre científico del pez.
sexo Sexo del pez si es true es hembra y si es false es macho.

Clase Filtrador

Clase abstracta pública que extiende de Pez la cual representa un pez con un tipo de alimentación herbívora.

Métodos

Filtrador(String nombre, String nombreCientifico, boolean sexo)

protected Filtrador(String nombre, String nombreCientifico, boolean sexo)
Constructor de la clase filtrador el cual llama a los parámetros de la superclase.

Parámetros

nombre Nombre del pez.
nombreCientifico Nombre científico del pez.
sexo Sexo del pez si es true es hembra y si es false es macho.

comer()

public int comer()
Método que devuelve la cantidad de comida que consumirá el pez.

devuelve La cantidad de comida que consumirá.

Clase Omnívoro

Clase abstracta pública que extiende de Pez la cual representa un pez con un tipo de alimentación omnívora.

Métodos

Omnivoro(String nombre, String nombreCientifico, boolean sexo)

protected Omnivoro(String nombre, String nombreCientifico, boolean sexo)
Constructor de la clase omnívoro el cual llama al constructor de la superclase.

Parámetros

nombre Nombre del pez.
nombreCientifico Nombre científico del pez.
sexo Sexo del pez si es true es hembra y si es false es macho.

comer()

public int comer()
Método que devuelve la cantidad de comida que consumirá el pez.

devuelve La cantidad de comida que consumirá.

Clase Caballa

Clase pública que hereda de la clase abstracta Carnivoro, implementa la interfaz bandera Mar y representa a una caballa.

Métodos

Caballa(boolean sexo)

public Caballa(boolean sexo)
Constructor de caballas. Este llama al constructor de la superclase.

Parámetros

sexo Sexo de la caballa si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado de la caballa. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que la caballa crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si la caballa está madura.

devuelve True si la caballa está madura.

isEdadOptima()

public boolean isEdadOptima()
Indica si la caballa está en la edad óptima para ser vendida.

devuelve True si la caballa está en la edad óptima para ser vendida.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve una caballa recién nacida de sexo masculino.

devuelve Caballa recién nacida de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve una caballa recién nacida de sexo femenino.

devuelve Caballa recién nacida de sexo femenino.

Clase Pejerrey

Clase pública que hereda de la clase abstracta Carnivoro, implementa la interfaz bandera Rio y representa a un pejerrey.

Métodos

Pejerrey(boolean sexo)

public Pejerrey(boolean sexo)
Constructor de pejerreyes. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del pejerrey si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del pejerrey. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el pejerrey crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el pejerrey está maduro

devuelve True si el pejerrey está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el pejerrey está en la edad óptima para ser vendido.

devuelve True si el pejerrey está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un pejerrey recién nacido de sexo masculino.

devuelve Pejerrey recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un pejerrey recién nacido de sexo femenino.

devuelve Pejerrey recién nacido de sexo femenino.

Clase PercaEuropea

Clase pública que hereda de la clase abstracta Carnivoro, implementa la interfaz bandera Rio y representa a una perca europea.

Métodos

PercaEuropea(boolean sexo)

public PercaEuropea(boolean sexo)
Constructor de percas europeas. Este llama al constructor de la superclase.

Parámetros

sexo Sexo de la perca europea si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado de la perca europea. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que la perca europea crezca un día. Este método sobrescribe el método abstracto de la superclase.

comer()

public int comer()
Indica la cantidad de alimento que consume la perca europea en un día.

devuelve Cantidad de alimento que consume la perca europa en un día.

isMaduro()

public boolean isMaduro()
Indica si la perca europea está madura.

devuelve True si la perca europea está madura.

isEdadOptima()

public boolean isEdadOptima()
Indica si la perca europea está en la edad óptima para ser vendida.

devuelve True si la perca europea está en la edad óptima para ser vendida.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve una perca europea recién nacida de sexo masculino.

devuelve Perca europea recién nacida de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve una perca europea recién nacida de sexo femenino.

devuelve Perca europea recién nacida de sexo femenino.

Clase Robalo

Clase pública que hereda de la clase abstracta Carnivoro, implementa la interfaz bandera Mar y representa a un róbalo.

Métodos

Robalo(boolean sexo)

public Robalo(boolean sexo)
Constructor de róbalos. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del róbalo si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del róbalo. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el róbalo crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el róbalo está maduro.

devuelve True si el róbalo está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el róbalo está en la edad óptima para ser vendido.

devuelve True si el róbalo está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un róbalo recién nacido de sexo masculino.

devuelve Róbalo recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un róbalo recién nacido de sexo femenino.

devuelve Róbalo recién nacido de sexo femenino.

Clase SalmonAtlantico

Clase pública que hereda de la clase abstracta Carnivoro, implementa las interfaces bandera Mar y Rio y representa a un salmón atlántico.

Métodos

SalmonAtlantico(boolean sexo)

public SalmonAtlantico(boolean sexo)
Constructor de salmones atlánticos. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del salmón atlántico si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del salmón atlántico. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el salmón atlántico crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el salmón atlántico está maduro.

devuelve True si el salmón atlántico está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el salmón atlántico está en la edad óptima para ser vendido.

devuelve True si el salmón atlántico está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un salmón atlántico recién nacido de sexo masculino.

devuelve Salmón atlántico recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un salmón atlántico recién nacido de sexo femenino.

devuelve Salmón atlántico recién nacido de sexo femenino.

Clase SalmonChinook

Clase pública que hereda de la clase abstracta Carnivoro, implementa la interfaz bandera Rio y representa a un salmón chinook.

Métodos

SalmonChinook(boolean sexo)

public SalmonChinook(boolean sexo)
Constructor de salmones chinook. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del salmón chinook si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del salmón chinook. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el salmón chinook crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el salmón chinook está maduro.

devuelve True si el salmón chinook está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el salmón chinook está en la edad óptima para ser vendido.

devuelve True si el salmón chinook está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un salmón chinook recién nacido de sexo masculino.

devuelve Salmón chinook recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un salmón chinook recién nacido de sexo femenino.

devuelve Salmón chinook recién nacido de sexo femenino.

Clase ArenqueDelAtlantico

Clase pública que hereda de la clase abstracta Filtrador, implementa la interfaz bandera Mar y representa a un arenque del atlántico.

Métodos

ArenqueDelAtlantico(boolean sexo)

public ArenqueDelAtlantico(boolean sexo)
Constructor de arenques del atlántico. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del arenque del atlántico si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del arenque del atlántico. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el arenque del atlántico crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el arenque del atlántico está maduro.

devuelve True si el arenque del atlántico está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el arenque del atlántico está en la edad óptima para ser vendido.

devuelve True si el arenque del atlántico está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un arenque del atlántico recién nacido de sexo masculino.

devuelve Arenque del atlántico recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un arenque del atlántico recién nacido de sexo femenino.

devuelve Arenque del atlántico recién nacido de sexo femenino.

Clase TilapiaDelNilo

Clase pública que hereda de la clase abstracta Filtrador, implementa la interfaz bandera Rio y representa a una tilapia del nilo.

Métodos

TilapiaDelNilo(boolean sexo)

public TilapiaDelNilo(boolean sexo)
Constructor de tilapias del nilo. Este llama al constructor de la superclase.

Parámetros

sexo Sexo de la tilapia del nilo si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado de la tilapia del nilo. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que la tilapia del nilo crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si la tilapia del nilo está madura.

devuelve True si la tilapia del nilo está madura.

isEdadOptima()

public boolean isEdadOptima()
Indica si la tilapia del nilo está en la edad óptima para ser vendida.

devuelve True si la tilapia del nilo está en la edad óptima para ser vendida.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve una tilapia del nilo recién nacida de sexo masculino.

devuelve Tilapia del nilo recién nacida de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve una tilapia del nilo recién nacida de sexo femenino.

devuelve Tilapia del nilo recién nacida de sexo femenino.

Clase Abadejo

Clase pública que hereda de la clase abstracta Omnivoro, implementa la interfaz bandera Mar y representa a un abadejo.

Métodos

Abadejo(boolean sexo)

public Abadejo(boolean sexo)
Constructor de abadejos. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del abadejo si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del abadejo. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el abadejo crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el abadejo está maduro.

devuelve True si el abadejo está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el abadejo está en la edad óptima para ser vendido.

devuelve True si el abadejo está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un abadejo recién nacido de sexo masculino.

devuelve Abadejo recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un abadejo recién nacido de sexo femenino.

devuelve Abadejo recién nacido de sexo femenino.

Clase CarpinTresEspinas

Clase pública que hereda de la clase abstracta Omnivoro, implementa la interfaz bandera Rio y representa a un carpín de tres espinas.

Métodos

CarpinTresEspinas(boolean sexo)

public CarpinTresEspinas(boolean sexo)
Constructor de carpines de tres espinas. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del carpín de tres espinas si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del capín de tres espinas. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el carpín de tres espinas crezca un día. Este método sobrescribe el método abstracto de la superclase.

comer()

public int comer()
Indica la cantidad de comida que consume el carpín de tres espinas un determinado día.

devuelve Cantidad de comida que consume el carpín de tres espinas un determinado día.

isMaduro()

public boolean isMaduro()
Indica si el carpín de tres espinas está maduro.

devuelve True si el carpín de tres espinas está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el carpín de tres espinas está en la edad óptima para ser vendido.

devuelve True si el capín de tres espinas está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un carpín de tres espinas recién nacido de sexo masculino.

devuelve Carpín de tres espinas recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un carpín de tres espinas recién nacido de sexo femenino.

devuelve Carpín de tres espinas recién nacido de sexo femenino.

Clase Dorada

Clase pública que hereda de la clase abstracta Omnivoro, implementa las interfaces bandera Rio y Mar y representa a una dorada.

Métodos

Dorada(boolean sexo)

public Dorada(boolean sexo)
Constructor de doradas. Este llama al constructor de la superclase.

Parámetros

sexo Sexo de la dorada si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado de la dorada. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que la dorada crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si la dorada está madura.

devuelve True si la dorada está madura.

isEdadOptima()

public boolean isEdadOptima()
Indica si la dorada está en la edad óptima para ser vendida.

devuelve True si la dorada está en la edad óptima para ser vendida.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve una dorada recién nacida de sexo masculino.

devuelve Dorada recién nacida de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve una dorada recién nacida de sexo femenino.

devuelve Dorada recién nacida de sexo femenino.

Clase Sargo

Clase pública que hereda de la clase abstracta Omnivoro, implementa la interfaz bandera Mar y representa a un sargo.

Métodos

Sargo(boolean sexo)

public Sargo(boolean sexo)
Constructor de sargos. Este llama al constructor de la superclase.

Parámetros

sexo Sexo del sargo si es true es hembra y si es false es macho.

showStatus()

public void showStatus()
Muestra el estado del sargo. Este método sobrescribe el método abstracto de la superclase.

grow()

public void grow()
Hace que el sargo crezca un día. Este método sobrescribe el método abstracto de la superclase.

isMaduro()

public boolean isMaduro()
Indica si el sargo está maduro.

devuelve True si el sargo está maduro.

isEdadOptima()

public boolean isEdadOptima()
Indica si el sargo está en la edad óptima para ser vendido.

devuelve True si el sargo está en la edad óptima para ser vendido.

obtenerPezHijo()

public Pez obtenerPezHijo()
Devuelve un sargo recién nacido de sexo masculino.

devuelve Sargo recién nacido de sexo maculino.

obtenerPezHija()

public Pez obtenerPezHija()
Devuelve un sargo recién nacido de sexo femenino.

devuelve Sargo recién nacido de sexo femenino.

Clase AlmacenCentral

Clase pública que representa a un almacén central que almacena comida de dos tipos (animal y vegetal) y la distribuye equitativamente entre las piscifactorías.

Atributos

private int capacidadComida

Capacidad máxima de comida para cada tipo del almacén central.

private int cantidadComidaAnimal

Cantidad de comida animal disponible en el almacén central.

private int cantidadComidaVegetal

Cantidad de comida vegetal disponible en el almacén central.

private boolean disponible

Indica si el almacén central está disponible.

Métodos

AlmacenCentral()

public AlmacenCentral()
Constructor de almacenes centrales que inician con una capacidad de 200 para comida animal y vegetal.

getCapacidadComida()

public int getCapacidadComida()
Permite obtener la capacidad máxima de comida para cada tipo del almacén central.

devuelve Capacidad máxima de comida para cada tipo del almacén central.

setCapacidadComida(int capacidadComida)

public void setCapacidadComida(int capacidadComida)
Permite establecer la capacidad máxima de cada tipo de comida del almacén central.

Parámetros
capacidadComida Capacidad máxima de cada tipo de comida a establecer.

getCantidadComidaAnimal()

public int getCantidadComidaAnimal()
Permite obtener la cantidad de comida animal disponible en el almacén central.

devuelve Cantidad de comida animal disponible en el almacén central.

setCantidadComidaAnimal(int cantidadComidaAnimal)

public void setCantidadComidaAnimal(int cantidadComidaAnimal)
Permite establecer la cantidad de comida animal disponible en el almacén central.

Parámetros
cantidadComidaAnimal Cantidad de comida animal disponible a establecer.

getCantidadComidaVegetal()

public int getCantidadComidaVegetal()
Permite obtener la cantidad de comida vegetal disponible en el almacén central.

devuelve Cantidad de comida vegetal disponible en el almacén central.

setCantidadComidaVegetal(int cantidadComidaVegetal)

public void setCantidadComidaVegetal(int cantidadComidaVegetal)
Permite establecer la cantidad de comida vegetal disponible en el almacén central.

Parámetros
cantidadComidaVegetal Cantidad de comida vegetal disponible a establecer.

isDisponible()

public boolean isDisponible()
Indica si el AlmacenCentral está disponible.

devuelve Si el AlmacenCentral está disponible o no.

setDisponible(boolean disponible)

public void setDisponible(boolean disponible)
Permite establecer la disponibilidad del AlmacenCentral.

Parámetros
disponible Disponibilidad del AlmacenCentral a establecer.

mejorar()

public void mejorar()
Mejora el almacén central aumentando la capacidad de ambas comidas en 50 unidades.

toString()

public String toString()
Devuelve un string con información relevante del almacén central. Este método sobrescribe al método de la superclase Object.

devuelve Información relevante del almacén central.

Clases

AdaptadorJSONAlmacenCentral

Clase privada que implementa las interfaces JsonSerializer<AlmacenCentral> y JsonDeserializer<AlmacenCentral> que se encarga de adaptar la serialización y deserialización de un objeto AlmacenCentral al formato json.

Métodos

deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context)

public AlmacenCentral deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context) throws JsonParseException
Se encarga de la deserialización de un objeto AlmacenCentral.

Parámetros

json Elemento json que contiene la información del objeto AlmacenCentral.
typeOfT Tipo del elemento a deserializar.
context Contexto de la deserialización.

devuelve Objeto AlmacenCentral deserializado.

lanza JsonParseException Cuando no se puede deserializar el objeto AlmacenCentral.

serialize(AlmacenCentral src, Type typeOfSrc, JsonSerializationContext context)

public JsonElement serialize(AlmacenCentral src, Type typeOfSrc, JsonSerializationContext context)

Parámetros

src Objeto AlmacenCentral a serializar.
typeOfSrc Tipo del objeto a serializar.
context Contexto de serialización.

devuelve Elemento json con la información del objeto AlmacenCentral serializada.

Clase Tanque

Clase pública que representa a un tanque que contiene un número de peces. Este hace uso de la clase Pez y de la clase pública estática interna AlmacenComida de la clase Piscifactoria.

Atributos

private final int numeroTanque

Número del tanque de la piscifactoría.

private transient int capacidadMaximaPeces

Capacidad máxima de peces que puede tener el tanque.

private ArrayList<Pez> peces

Peces del tanque.

Métodos

getCapacidadMaximaPeces()

public int getCapacidadMaximaPeces()
Permite obtener la capacidad máxima de peces que puede tener el tanque.

devuelve Capacidad máxima de peces que puede tener el tanque.

setCapacidadMaximaPeces(int capacidadMaximaPeces)

public void setCapacidadMaximaPeces(int capacidadMaximaPeces)
Permite establecer la capacidad máxima de peces que puede tener el tanque.

Parámetros

capacidadMaximaPeces Capacidad máxima de peces que puede tener el tanque a establecer.

getPeces()

public ArrayList<Pez> getPeces()
Permite obtener los peces del tanque.

devuelve Peces del tanque.

setPeces(ArrayList<Pez> peces)

public void setPeces(ArrayList<Pez> peces)
Permite establecer los peces del tanque.

Parámetros

peces Peces del tanque a establecer.

getNumeroTanque()

public int getNumeroTanque()
Permite obtener el número del tanque en la piscifactoría.

devuelve Número del tanque en la piscifactoría.

Tanque(int numeroTanque, int capacidadMaximaPeces)

public Tanque(int numeroTanque, int capacidadMaximaPeces)
Constructor de tanques.

Parámetros
numeroTanque Número del tanque en la piscifactoría.
capacidadMaximaPeces Capacidad máxima de peces del tanque.

showStatus()

public void showStatus()
Imprime el estado del tanque por pantalla.

pecesVivos()

public int pecesVivos()
Devuelve el número de peces vivos en el tanque.

devuelve Número de peces vivos en el tanque.

pecesAlimentados()

public int pecesAlimentados()
Devuelve el número de peces alimentados en el tanque.

devuelve Número de peces alimentados en el tanque.

pecesAdultos()

public int pecesAdultos()
Devuelve el número de peces adultos en el tanque.

devuelve Número de peces adultos en el tanque.

pecesAdultosVivos()

public int pecesAdultosVivos()
Devuelve el número de peces adultos vivos en el tanque.

devuelve Número de peces adultos vivos en el tanque.

pecesMacho()

public int pecesMacho()
Devuelve el número de peces macho en el tanque.

devuelve Número de peces macho en el tanque.

pecesHembra()

public int pecesHembra()
Devuelve el número de peces hembra en el tanque.

devuelve Número de peces hembra en el tanque.

pecesFertiles()

public int pecesFertiles()
Devuelve el número de peces fértiles en el tanque.

devuelve Número de peces fértiles en el tanque.

showFishStatus()

public void showFishStatus()
Muestra el estado de cada pez del tanque por pantalla.

showCapacity(String piscifactoria)

public void showCapacity(String piscifactoria)
Imprime el porcentaje de llenado del tanque por pantalla.

Parámetros
piscifactoria Nombre de la piscifactoría en la que se sitúa el tanque.

alimentar(Piscifactoria.AlmacenComida almacenComida)

public void alimentar(Piscifactoria.AlmacenComida almacenComida)
Gestiona la lógica para alimentar a los peces.

Parámetros
almacenComida Almacén de comida de la piscifactoría donde se sitúa el tanque.

alimentarAleatorio(ArrayList<Integer> cantidadDeComidaNecesariaPorPez, Piscifactoria.AlmacenComida almacenComida, int comidaDisponible)

private void alimentarAleatorio(ArrayList<Integer> cantidadDeComidaNecesariaPorPez, Piscifactoria.AlmacenComida almacenComida, int comidaDisponible)
Gestiona la lógica de alimentación de los peces cuando la comida es insuficiente para alimentar a todos los peces.

Parámetros

cantidadDeComidaNecesariaPorPez Cantidad de comida que necesita cada pez para alimentarse.
almacenComida Almacén de comida de la piscifactoría donde se sitúa el tanque.
comidaDisponible Comida de la que se dispone para alimentar a los peces.

hayMachoFertil()

private boolean hayMachoFertil()
Indica si hay un macho fértil en la piscifactoría.

devuelve True si hay un macho fértil en la piscifactoría.

reproducir()

private void reproducir()
Gestiona la lógica de reproducción de los peces del tanque.

venderPecesOptimos()

private void venderPecesOptimos()
Vende todos los peces que se encuentran en la edad óptima para ser vendidos.

venderPeces()

public void venderPeces()
Vende todos los peces que hayan llegado a la madurez.

eliminarPecesMuertos()

public void eliminarPecesMuertos()
Elimina los peces muertos del tanque.

nextDay()

public int nextDay()
Realiza la lógica de que ha pasado un día haciendo crecer a los peces, reproduciendolos y vendiendo los peces que se encuentren en edad óptima.

devuelve Peces vendidos.

vaciarTanque()

public void vaciarTanque()

Elimina todos los peces de un tanque, independientemente de si están vivos o muertos y de su edad.

toString()

public String toString()
Devuelve un string con información relevante del tanque. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante del tanque.

Clases

AdaptadorJSON

Clase privada que implementa las interfaces JsonDeserializer<Tanque> y JsonSerializer<Tanque> que se encarga de adaptar la serialización y deserialización de un objeto Tanque al formato json.

Métodos

deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context)

public Tanque deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context) throws JsonParseException
Se encarga de la deserialización de un objeto Tanque.

Parámetros
json Elemento json que contiene la información del objeto Tanque.
typeOfT Tipo del elemento a deserializar.
context Contexto de la deserialización.

devuelve Objeto Tanque deserializado.

lanza JsonParseException Cuando no se puede deserializar el objeto Tanque.

serialize(Tanque src, Type typeOfSrc, JsonSerializationContext context)

public JsonElement serialize(Tanque src, Type typeOfSrc, JsonSerializationContext context)
Se encarga de la serialización de un objeto Tanque.

Parámetros
src Objeto Tanque a serializar.
typeOfSrc Tipo del objeto a serializar.
context Contexto de serialización.

devuelve Elemento json con la información del objeto Tanque serializada.

Clase Piscifactoría

Clase pública abstracta que representa a una piscifactoría con múltiples tanques de peces. Esta hace uso de la clase tanque.

Clases

AlmacenComida

Clase interna de la clase Piscifactoría que representa a un almacén de comida de una piscifactoría.

Atributos

private int capacidadMaximaComida

Capacidad máxima de cada tipo de comida en el almacén.

private int cantidadComidaAnimal

Cantidad de comida animal que hay en el almacén.

private int cantidadComidaVegetal

Cantidad de comida vegetal que hay en el almacén.

Métodos

AlmacenComida(int capacidadMaximaComida, int cantidadComidaAnimal, int cantidadComidaVegetal)

public AlmacenComida(int capacidadMaximaComida, int cantidadComidaAnimal, int cantidadComidaVegetal)
Constructor parametrizado de almacenes de comida.

Parámetros

capacidadMaximaComida Capacidad máxima de comida de cada tipo.
cantidadComidaAnimal Cantidad de comida animal del almacén.
cantidadComidaVegetal Cantidad de comida vegetal del almacén.

getCapacidadMaximaComida()

public int getCapacidadMaximaComida()
Permite obtener la capacidad máxima de comida de cada tipo del almacén.

devuelve Capacidad máxima de comida de cada tipo del almacén.

setCapacidadMaximaComida(int capacidadMaximaComida)

public void setCapacidadMaximaComida(int capacidadMaximaComida)
Permite establecer la capacidad máxima de comida de cada tipo del almacén.

Parámetros

capacidadMaximaComida Capacidad máxima de comida de cada tipo de almacén a establecer.

getCantidadComidaAnimal()

public int getCantidadComidaAnimal()
Permite obtener la cantidad de comida animal del almacén.

devuelve Cantidad de comida animal del almacén.

setCantidadComidaAnimal(int cantidadComidaAnimal)

public void setCantidadComidaAnimal(int cantidadComidaAnimal)
Permite establecer la cantidad de comida animal del almacén.

Parámetros

cantidadComidaAnimal Cantidad de comida animal del almacén a establecer.

getCantidadComidaVegetal()

public int getCantidadComidaVegetal()
Permite obtener la cantidad de comida vegetal del almacén.

devuelve Cantidad de comida vegetal del almacén.

setCantidadComidaVegetal(int cantidadComidaVegetal)

public void setCantidadComidaVegetal(int cantidadComidaVegetal)
Permite establecer la cantidad de comida vegetal del almacén.

Parámetros

cantidadComidaVegetal Cantidad de comida vegetal del almacén a establecer.

mejorar(int capacidadAAumentar)

public void mejorar(int capacidadAAumentar)
Mejora el almacén aumentando la capacidad máxima de comida de cada tipo.

Parámetros

capacidadAAumentar Unidades a aumentar la capacidad máxima de comida de cada tipo.

toString()

public String toString()
Devuelve un string con información relevante del almacén.

devuelve String con información relevante del almacén.

AdaptadorJSON

Clase privada que implementa las interfaces JsonDeserializer<Piscifactoria> y JsonSerializer<Piscifactoria> que se encarga de adaptar una objeto Piscifactoria para que pueda ser serializado y deserializado en formato json.

Métodos

serialize(Piscifactoria src, Type typeOfSrc, JsonSerializationContext context)

public JsonElement serialize(Piscifactoria src, Type typeOfSrc, JsonSerializationContext context)
Se encarga de la serialización de un objeto Piscifactoria.

Parámetros
src Objeto Piscifactoria a serializar.
typeOfSrc Tipo del objeto a serializar.
context Contexto de serialización.

devuelve Elemento json con la información del objeto Piscifactoria serializada.

deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context)

public Piscifactoria deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context) throws JsonParseException
Se encarga de la deserialización de un objeto Piscifactoria.

Parámetros
json Elemento json que contiene la información del objeto Piscifactoria.
typeOfT Tipo del objeto a deserializar.
context Contexto de deserialización.

devuelve Objeto Piscifactoria deserializado.

lanza JsonParseException Cuando no se puede deserializar el objeto Piscifactoria.

Atributos

protected ArrayList<Tanque> tanques

Tanques de la piscifactoría.

protected Tanque tanqueInicial

El tanque con el que empieza obligatoriamente la piscifactoría.

protected AlmacenComida almacenInicial

Almacén de comida de la piscifactoría.

protected String nombre

El nombre de la piscifactoría.

Métodos

Piscifactoria(String nombre)

protected Piscifactoria(String nombre)
Constructor de la clase Piscifactoría el cual establece el nombre de la piscifactoría.

Parámetros

nombre nombre de la piscifactoría.

getTanques()

public ArrayList<Tanque> getTanques()
Devuelve los tanques.

devuelve Tanques de la piscifactoría.

setTanques(ArrayList<Tanque> tanques)

public void setTanques(ArrayList<Tanque> tanques)
Permite establecer los tanques de la piscifactoría.

Parámetros

tanques Tanques a establecer.

getTanqueInicial()

public Tanque getTanqueInicial()
Devuelve el tanque inicial de la piscifactoría.

devuelve Tanque inicial de la piscifactoría.

setTanqueInicial(Tanque tanqueInicial)

public void setTanqueInicial(Tanque tanqueInicial)
Permite establecer el tanque inicial de la piscifactoría.

Parámetros
tanqueInicial Tanque inicial a establecer.

getNombre()

public String getNombre()
Devuelve el nombre de la piscifactoría.

devuelve Nombre de la piscifactoría.

getAlmacenInicial()

public AlmacenComida getAlmacenInicial()
Devuelve el almacén de comida de la piscifactoría.

devuelve Almacén de comida de la piscifactoría.

setAlmacenInicial(AlmacenComida almacenInicial)

public void setAlmacenInicial(AlmacenComida almacenInicial)
Permite establecer el almacén de la piscifactoría.

Parámetros
almacenInicial Almacén inicial de la piscifactoría a establecer.

setNombre(String nombre)

public void setNombre(String nombre)
Permite establecer el nombre de la piscifactoría.

Parámetros
nombre Nombre de la piscifactoría a establecer.

showStatus()

public void showStatus()
Imprime el estado de la piscifactoría.

datosTanques()

private String datosTanques()
Devuelve el resto del texto, datos y cálculos de showStatus().

> **devuelve** Texto y cálculos necesarios para mostrar el estado de la piscifactoría en el método showStatus().

showTankStatus()

public void showTankStatus()
Imprime el estado de los tanques de la piscifactoría.

showFishStatus()

public void showFishStatus()
Imprime el estado de los peces de los tanques.

showCapacity()

public void showCapacity()
Imprime la capacidad de los tanques.

showFood()

public void showFood()
Imprime el estado del almacén

sellFish()

public void sellFish()
Vende todos los peces maduros y vivos de los tanques.

upgradeFood()

public abstract void upgradeFood()
Mejora el almacén e imprime en qué cantidad se ha mejorado.

nextDay()

public int nextDay()
Pasa un día en la piscifactoría realizando toda la lógica relacionada.

> **devuelve** Número de peces vendidos cuando pasa el día.

getPecesVivos()

public int getPecesVivos()
Devuelve el número de peces vivos en la piscifactoría.

> **devuelve** Número de peces vivos.

getPecesTotales()

public int getPecesTotales()
Devuelve el número de peces totales de la piscifactoría.

devuelve Número de peces totales.

getEspacioPeces()

public int getEspacioPeces()
Devuelve el espacio total de peces en la piscifactoría.

devuelve Espacio total de peces en la piscifactoría.

getIndiceTanqueVacio()

public int getIndiceTanqueVacio()
Permite obtener el índice del ArrayList de un tanque vacío de la piscifactoría.

devuelve Índice del ArrayList de un tanque vacío de la piscifactoría o -1 sino hay ningún tanque vacío.

getIndiceTanqueConEspacioParaPez(String nombrePez)

public int getIndiceTanqueConEspacioParaPez(String nombrePez)
Permite obtener el índice del ArrayList de un tanque que tiene un espacio para un pez en concreto.

Parámetros

nombrePez Nombre del pez del cual se quiere obtener el índice del tanque que tiene espacio para él.

devuelve Índice del tanque que tiene un espacio para un pez en concreto o -1 si no hay un tanque con un espacio para el pez en concreto.

isTodosLosTanqueLlenos()

public boolean isTodosLosTanqueLlenos()
Indica si todos los tanques de la piscifactoría están llenos.

devuelve True si todos los tanques de la piscifactoría están llenos.

toString()

public String toString()
Devuelve un string con información relevante de la piscifactoría. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante de la piscifactoría.

Clase PiscifactoriaRio

Clase pública que representa a una piscifactoría de río y que hereda de la clase Piscifactoria.

Métodos

PiscifactoriaRio()

public PiscifactoriaRio()
Constructor de piscifactorías de río sin parámetros. Este hace uso del constructor de la superclase.

PiscifactoriaRio(String nombre)

public PiscifactoriaRio(String nombre)
Constructor de piscifactorías de río el cual establece el nombre de la piscifactoría. Este hace uso del constructor de la superclase,

Parámetros

nombre Nombre de la piscifactoría de río.

upgradeFood()

public void upgradeFood()
Mejora el almacén de comida aumentando en 25 unidades la capacidad máxima e imprime en qué cantidad se ha mejorado.

toString()

public String toString()
Devuelve un string con información relevante de la piscifactoría de río. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante de la piscifactoría de río.

Clase PiscifactoriaMar

Clase pública que representa a una piscifactoría de mar y que hereda de la clase Piscifactoria.

Métodos

PiscifactoriaMar()

public PiscifactoriaMar()
Constructor de piscifactorías de mar sin parámetros. Este hace uso del constructor de la superclase.

PiscifactoriaMar(String nombre)

public PiscifactoriaMar(String nombre)
Constructor de piscifactorías de mar el cual establece el nombre de la piscifactoría. Esta hace uso del constructor de la superclase.

Parámetros

nombre Nombre de la piscifactoría de mar.

upgradeFood()

public void upgradeFood()
Mejora el almacén de comida aumentando en 100 unidades la capacidad máxima e imprime en qué cantidad se ha mejorado.

toString()

public String toString()
Devuelve un string con información relevante de la piscifactoría de mar. Este método sobrescribe al método de la superclase Object.

devuelve String con información relevante de la piscifactoría de mar.

Clase Simulador

Clase pública que representa a un simulador de piscifactoría esta hace uso de todas las otras clases del sistema y contiene el main.

Clases

AdaptadorJSONEstadisticas

Clase privada que implementa las interfaces JsonSerializer<Estadisticas> y JsonDeserializer<Estadisticas> que se encarga de adaptar la serialización y deserialización de un objeto Estadisticas al formato json.

Métodos

deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context)

public Estadisticas deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext context) throws JsonParseException
Se encarga de la deserialización de un objeto Estadisticas.

Parámetros

json Elemento json que contiene la información del objeto Estadisticas.
typeOfT Tipo del objeto a deserializar.
context Contexto de deserialización.

devuelve Objeto Estadisticas deserializado.

lanza JsonParseException Cuando el objeto Estadisticas no se puede deserializar.

serialize(Estadisticas src, Type typeOfSrc, JsonSerializationContext context)

public JsonElement serialize(Estadisticas src, Type typeOfSrc, JsonSerializationContext context)
Se encarga de la serialización de un objeto Estadisticas.

Parámetros

src Objeto Estadisticas a serializar.
typeOfSrc Tipo del objeto a serializar.
context Contexto de serialización.

devuelve Elemento json con la información del objeto Estadisticas serializada.

Atributos

public String[] pecesImplementados

@SerializedName(“implementados”)

public String[] pecesImplementados
Peces implementados en la simulación.

private int diasPasados

@SerializedName(“dia”)
private int diasPasados
Indica el número de días que han pasado en la simulación.

public ArrayList<Piscifactoria> piscifactorias

Piscifactoría de las que se dispone en la simulación.

private String nombre

@SerializedName(“empresa”)
private String nombre
Nombre de la entidad, empresa o partida de la simulación.

public SistemaMonedas sistemaMonedas

@SerializedName(“monedas”)
public SistemaMonedas sistemaMonedas
Sistema de monedas usado en la simulación.

public AlmacenCentral almacenCentral

@SerializedName(“edificios”)
public AlmacenCentral almacenCentral
Almacén central de comida usado en la simulación.

public Estadisticas estadisticas

@SerializedName(“orca”)
@JsonAdapter(AdaptadorJSONEstadisticas.class)
public Estadisticas estadisticas
Estadísticas de los peces de la simulación.

public static final File archivoLogsGeneral

Archivo de *logs* donde se registran todos los errores que hayan sucedido en cualquier partida.

public static File archivoLogPartida

Archivo de *log* de la partida.

public static File archivoTranscripcionesPartida

Archivo de transcripciones de la partida.

public static File archivoGuardadoPartida

Archivo de guardado de la partida.

public static Simulador simulador

Simulador que se ejecuta en la partida.

Métodos

init()

private static void init()
Inicializa la simulación con unos datos solicitados al usuario.

menu()

private static void menu()
Imprime por pantalla el menú principal.

menuPisc()

private static void menuPisc()
Imprime por pantalla un menú para seleccionar una piscifactoría de la simulación.

selectPisc()

private static int selectPisc()
Permite seleccionar una piscifactoría del menú de piscifactorías.

devuelve Opción seleccionada.

selectTank(Piscifactoria piscifactoria)

private static int selectTank(Piscifactoria piscifactoria)
Muestra el menú de tanque de una piscifactoría y permite al usuario seleccionar uno.

Parámetros

piscifactoria Piscifactoría de la cuál se muestra el menú del tanque.

devuelve Índice del tanque seleccionado.

showGeneralStatus()

private static void showGeneralStatus()
Imprime por pantalla el estado general de la simulación.

showSpecificStatus()

private static void showSpecificStatus()
Muestra por pantalla el estado de todos los tanques de una piscifactoría seleccionada por el usuario.

showTankStatus(Piscifactoria piscifactoria)

private static void showTankStatus(Piscifactoria piscifactoria)

Muestra un menú para que el usuario seleccione el tanque de una piscifactoría y muestra el estado del tanque seleccionado.

Parámetros

piscifactoria Piscifactoría de la que se muestra el estado del tanque.

showStats()

private static void showStats()
Muestra un desglose de las estadísticas por cada tipo de pez.

showIctio()

private static void showIctio()
Muestra la información relativa a un pez seleccionado por el usuario.

nextDay()

private static void nextDay()
Avanza un día en todas las piscifactorías y muestra el número de peces vendidos y las monedas ganadas con ello.

addFood()

private static void addFood()
Gestiona la lógica de compra de comida para el almacén central o para una piscifactoría.

menuTipoComida()

private static int menuTipoComida()
Muestra un menú para que el usuario escoja el tipo de comida.

devuelve Opción seleccionada por el usuario.

menuCantidadComida()

private static int menuCantidadComida()
Muestra un menú para seleccionar la cantidad de comida a añadir.

devuelve Opción seleccionada por el usuario.

calcularCosto(int cantidad)

private static int calcularCosto(int cantidad)
Calcula el costo de la comida a añadir, aplicando descuentos cada 25 unidades.

Parámetros

cantidad Cantidad de comida a añadir.

devuelve Costo total en monedas.

addFish()

private static void addFish()
Añade un pez a una piscifactoría seleccionada por el usuario.

addFishMar(Piscifactoria piscifactoria)

private static void addFishMar(Piscifactoria piscifactoria)
Gestiona la lógica de añadir un pez a una piscifactoría de mar.

Parámetros

piscifactoria Piscifactoría donde se va a añadir el pez.

addFishRio(Piscifactoria piscifactoria)

private static void addFishRio(Piscifactoria piscifactoria)
Gestiona la lógica de añadir un pez a una pisicfactoría de río.

Parámetros

piscifactoria Piscifactoría donde se va a añadir el pez.

crearPezMar(int pez, boolean sexo)

private static Pez crearPezMar(int pez, boolean sexo)
Crea un pez que puede vivir en una piscifactoría de mar.

Parámetros

pez Código numérico del pez a crear.
sexo Sexo del pez a crear.

devuelve Pez que puede vivir en una piscifactoría de mar creado.

crearPezRio(int pez, boolean sexo)

private static Pez crearPezRio(int pez, boolean sexo)
Crea un pez que puede vivir en una piscifactoría de río.

Parámetros

pez Código numérico del pez a crear.
sexo Sexo del pez a crear.

devuelve Pez que puede vivir en una piscifactoría de río creado.

mostrarInformacionPez(PecesDatos datosDelPez)

private static void mostrarInformacionPez(PecesDatos datosDelPez)
Imprime por pantalla los datos relativos a un pez.

Parámetros

datosDelPez Datos relativos a un pez a mostrar.

sell()

private static void sell()
Vende todos los peces adultos que estén vivos en una piscifactoría seleccionada.

cleanTank()

private static void cleanTank()
Elimina todos los peces muertos de una piscifactoría seleccionada.

emptyTank()

private static void emptyTank()
Elimina todos los peces de un tanque, estén vivos o muertos.

upgrade()

private static void upgrade()
Muestra un menú para hacer mejoras como comprar o mejorar un edificio.

comprarEdificio()

private static void comprarEdificio()
Muestra un menú para comprar edificios.

comprarPiscifactoria()

private static void comprarPiscifactoria()
Permite al usuario comprar una nueva piscifactoría.

mejorarEdificio()

private static void mejorarEdificio()
Muestra un menú para mejorar edificios existentes.

mejorarPiscifactoria()

private static void mejorarPiscifactoria()
Permite al usuario mejorar una piscifactoría seleccionada.

mejorarAlmacenComidaPiscifactoria(Piscifactoria piscifactoria)

private static void mejorarAlmacenComidaPiscifactorio(Piscifactoria piscifactoria)
Gestiona la lógica de mejorar el almacén de comida de una piscifactoría.

Parámetros

piscifactoria Piscifactoría de la que se mejora el almacén de comida.

comprarTanque(Piscifactoria piscifactoria)

private static void comprarTanque(Piscifactoria piscifactoria)
Permite al usuario comprar un tanque para una piscifactoría.

Parámetros

piscifactoria Piscifactoría en la que se compra el tanque.

aumentarCapacidadAlmacenCentral()

private static void aumentarCapacidadAlmacenCentral()
Aumenta la capacidad del almacén central.

seleccionarTipoPiscifactoría()

private static int seleccionarTipoPiscifactoria()
Muestra un menú para seleccionar el tipo de piscifactoría

devuelve Opción seleccionada por el usuario.

calcularCostoPiscifactoría(int tipo)

private static int calcularCostoPiscifactoria(int tipo)
Calcula el costo de adquirir una piscifactoría en función de si es de mar o de río.

Parámetros
tipo Tipo de la piscifactoría si es de mar o de río.

devuelve Costo de adquirir la piscifactoría.

numeroPiscifactoriasRio()

private static int numeroPiscifactoriaRio()
Permite obtener el número de piscifactorías de río de la simulación.

devuelve Número de piscifactorías de río de la simulación.

numeroPiscifactoriasMar()

private static int numeroPiscifactoriaMar()
Permite obtener el número de piscifactorías de mar de la simulación.

devuelve Número de piscifactorías de mar de la simulación.

mostrarEstadoTanque()

private static void mostrarEstadoTanque()
Gestiona la lógica para mostrar un tanque de una piscifactoría seleccionada por el usuario.

pasarDias()

private static void pasarDias()
Método que permite al usuario pasar varios días en la simulación.

anadirPezAleatorio()

private static void anadirPezAleatorio()
Añade 4 peces aleatorios a una piscifactoría seleccionada por el usuario.

repartirComida()

private static void repartirComida()
Reparte la comida del almacén central equitativamente entre las piscifactorías.

todasLasPiscifactoriasEnLaMediaComidaAnimal(int mediaCantidadComidaAnimal)

private static boolean todasLasPiscifactoriasEnLaMediaComidaAnimal(int mediaCantidadComidaAnimal)
Indica si todas las piscifactorías que no están llenas están en la media en cuanto a capacidad de comida animal.

Parámetros

mediaCantidadComidaAnimal Media de la cantidad de comida animal.

devuelve True si todas las piscifactorías que no están llenas están en la media en cuanto a cantidad de comida animal.

todasLasPiscifactoriasLlenasDeComidaAnimal()

private static boolean todasLasPiscifactoriasLlenasDeComidaAnimal()
Indica si todas las piscifactorías están llenas de comida animal.

devuelve True si todas las piscifactorías están llenas de comida animal.

mediaComidaAnimal()

private static int mediaComidaAnimal()
Devuelve la media de comida animal de las piscifactorías que no estén llenas.

devuelve Media de comida animal de las piscifactorías que no estén llenas.

repartirComidaAnimal()

private static void repartirComidaAnimal()
Gestiona la lógica de distribución equitativa de comida animal del almacén central a las piscifactorías.

todasLasPiscifactoriasEnLaMediaComidaVegetal(int mediaCantidadComidaVegetal)

private static boolean todasLasPiscifatoriasEnLaMediaComidaVegetal(int mediaCantidadComidaVegetal)
Indica si todas las piscifactorías que no están llenas están en la media en cuanto a cantidad de comida vegetal.

Parámetros

mediaCantidadComidaVegetal Media de la cantidad de comida vegetal

devuelve True si todas las piscifactorías que no están llenas están en la media en

cuanto a cantidad de comida vegetal.

todasLasPiscifactoriasLlenasDeComidaVegetal()

private static boolean todasLasPiscifactoriasLlenasDeComidaVegetal()
Indica si todas las piscifactorías están llenas de comida vegetal.

devuelve True si todas las piscifactorías están llenas de comida vegetal.

mediaComidaVegetal()

private static int mediaComidaVegetal()
Devuelve la media de comida vegetal de las piscifactorías que no estén llenas.

devuelve Media de comida vegetal de las piscifactorías que no estén llenas.

repartirComidaVegetal()

private static void repartirComidaVegetal()
Gestiona la lógica de distribución equitativa de la comida vegetal del almacén central a las piscifactorías.

anadirMonedasOculto()

private static void anadirMonedasOculto()
Añade 1000 monedas al sistema de monedas de la simulación.

toString()

public String toString()
Devuelve un string con información relevante de la clase.

devuelve String con información relevante de la clase.

main(String[ ] args)

public static void main(String[ ] args)
Método principal del programa que gestiona el uso del programa por parte del usuario. Este método sobrescribe al método de la superclase Object.

Parámetros
param Argumentos pasados por línea de comandos.

Clase Conexion

Clase pública que se encarga de abrir y cerrar la conexión con la base de datos.

Atributos

private static final String USER

Usuario con el que accede a la base de datos. Este atributo tiene como valor establecido “administrador_piscifactoria” que se trata de un usuario con permisos sobre la base de datos de la piscifactoría.

private static final String PASSWORD

Contraseña de acceso del usuario con el que se accede a la base de datos. Este atributo tiene como valor establecido “yu40/&23?Pl” ya que es la contraseña del usuario “administrador_piscifactoria”.

private static final String SERVER

Dirección del servidor donde está la base de datos siendo esta “ldominguez.iescotarelo.es”.

private static final int PORT

Puerto de escucha del sistema gestor de base de datos siendo este 3306 el predeterminado para MySQL.

private static Connextion conexion

Conexión con la base de datos.

Métodos

getConexion()

public static Connection getConexion()
Permite la obtención de la conexión con la base de datos.

> **devuelve** Conexión con la base de datos.

close()

public static void close()
Cierra la conexión con la base de datos.

Clase DAOPedidos

Clase pública DAO (Data Access Object) que gestiona el acceso a la base de datos de la simulación.

Atributos

private Connection conexion

Conexión con la base de datos.

private PreparedStatement consultaClientes

Consulta que obtiene todos los clientes de la base de datos.

private PreparedStatement consultaPeces

Consulta que obtiene todos los peces de la base de datos.

private PreparedStatement consultaPedidos

Consulta que obtiene todos los pedidos de la base de datos.

private PreparedStatement consultaPedidosNoRealizados

Consulta que recupera los datos de los pedidos que no se han cumplido al 100% con el nombre del cliente, el nombre del pedido, el porcentaje de completado y el número de referencia del pedido.

private PreparedStatement insercionCliente

Sentencia para la inserción de un cliente en la base de datos.

private PreparedStatement insercionPedido

Sentencia para la inserción de un pedido en la base de datos.

private PreparedStatement insercionPez

Sentencia para la inserción de un pez en la base de datos.

Métodos

DAOPedidos(Connection conexion)

public DAOPedidos(Connection conexion)
Constructor parametrizado que instancia un objeto de la clase a partir de una conexión con la base de datos.

Parámetros

conexion Conexión con la base de datos.

obtenerClientes()

public ArrayList<DTOCliente> obtenerClientes()
Realiza una consulta en la que se obtienen todos los clientes de la base de datos.

devuelve Clientes en la base de datos.

obtenerPeces()

public ArrayList<DTOPez> obtenerPeces()
Realiza una consulta en la que se obtienen todos los peces en la base de datos.

devuelve Peces en la base de datos.

obtenerPedidos()

public ArrayList<DTOPedido> obtenerPedidos()
Realiza una consulta en la que se obtienen todos los pedidos de la base de datos.

devuelve Pedidos en la base de datos.

borrarPedidos()

public void borrarPedidos()
Elimina todos los pedidos almacenados en la tabla pedidos.

obtenerPedidosNoFinalizados()

public ArrayList<DTOPedidoUsuarioPez> obtenerPedidosNoFinalizados()
Obtiene el número de referencia, el nombre del cliente, el nombre del pez y el porcentaje de completado de los pedidos que no han sido finalizados.

devuelve Número de referencia, nombre del cliente, nombre del pez y porcentaje de completado de los pedidos que no han sido finalizados.

insertarCliente(DTOCliente cliente)

public void insertarCliente(DTOCliente cliente)
Permite la inserción de un nuevo cliente en la base de datos.

Parámetros
cliente Cliente nuevo a insertar.

insertarPedido(DTOPedido pedido)

public void insertarPedido(DTOPedido pedido)
Permite la inserción de un nuevo pedido en la base de datos.

Parámetros
pedido Pedido nuevo a insertar.

insertarPez(DTOPez pez)

public void insertarPez(DTOPez pez)
Permite la inserción de un nuevo pez en la base de datos.

Parámetros

pez Pez nuevo a insertar.

close()

public void close()
Cierra la conexión a la base de datos y las consulta preparadas.

GeneradorBD

Clase pública encargada de crear la base de datos y de insertar los datos iniciales en ella.

Atributos

private static Connection conexion

Conexión con la base de datos.

Métodos

crearTablas()

public static void crearTablas()
Método que se encarga de crear todas las tablas de la base de datos.

insertarClientes()

public static void insertarClientes()
Inserta 10 clientes fijos en la base de datos.

insertarPeces()

public static void insertarPeces()
Inserta los datos de los peces implementados en la base de datos.

close()

public static void close()
Cierra la conexión con la base de datos.

DTOCliente

Clase pública que representa a un DTO de la tabla cliente.

Atributos

private int id

Id del cliente.

private String nombre

Nombre del cliente.

private String nif

Nif del cliente.

private String telefono

Teléfono del cliente.

Métodos

DTOCliente(int id, String nombre, String nif, String telefono)

public DTOCliente(int id, String nombre, String nif, String telefono)
Constructor parametrizado.

Parámetros

id Id del cliente.
nombre Nombre del cliente.
nif Nif del cliente.
telefono Teléfono del cliente.

DTOCliente(String nombre, String nif, String telefono)

public DTOCliente(String nombre, String nif, String telefono)
Constructor de DTOCliente sin id.

Parámetros

nombre Nombre del cliente.
nif Nif del cliente.
telefono Teléfono del cliente.

getId()

public int getId()
Permite obtener el id del cliente.

devuelve Id del cliente.

setId(int id)

public void setId(int id)
Permite establecer el id del cliente.

Parámetros
id Id del cliente a establecer.

getNombre()

public String getNombre()
Permite obtener el nombre del cliente.

devuelve Nombre del cliente.

setNombre(String nombre)

public void setNombre(String nombre)
Permite establecer el nombre del cliente.

Parámetros
nombre Nombre del cliente a establecer.

getNif()

public String getNif()
Permite obtener el nif del cliente.

devuelve Nif del cliente.

setNif(String nif)

public void setNif(String nif)
Permite establecer el nif del cliente.

Parámetros
nif Nif del cliente a establecer.

getTelefono()

public String getTelefono()
Permite obtener el teléfono del cliente.

devuelve Teléfono del cliente.

setTelefono(String telefono)

public setTelefono(String telefono)
Permite establecer el teléfono del cliente.

Parámetros
telefono Teléfono del cliente a establecer.

DTOPedido

Clase pública que representa a un DTO de la tabla pedido.

Atributos

private int numeroReferencia

Número de referencia del pedido.

private int idCliente

Id del cliente que realizó el pedido.

private int idPez

Id del pez del que se ha realizado el pedido.

private int pecesSolicitados

Número de peces solicitados en el pedido.

private int pecesEnviados

Número de peces enviados.

Métodos

DTOPedido(int numeroReferencia, int idCliente, int idPez, int pecesSolicitados, int pecesEnviados)

public DTOPedido(int numeroReferencia, int idCliente, int idPez, int pecesSolicitados, int pecesEnviados)
Constructor parametrizado.

Parámetros

numeroReferencia Número de referencia del pedido.
idCliente Id del cliente que realizó el pedido.
idPez Id del pez del que se ha realizado el pedido.
pecesSolicitados Número de peces solicitados en el pedido.
pecesEnviados Número de peces enviados.

DTOPedido(int idCliente, int idPez, int pecesSolicitados, int pecesEnviados)

public DTOPedido(int idCliente, int idPez, int pecesSolicitados, int pecesEnviados)
Constructor de pedidos sin número de referencia.

Parámetros

idCliente Id del cliente que realizó el pedido.
idPez Id del pez del que se ha realizado el pedido.
pecesSolicitados Número de peces solicitados en el pedido.
pecesEnviados Número de peces enviados.

getNumeroReferencia()

public int getNumeroReferencia()
Permite obtener el número de referencia del pedido.

devuelve Número de referencia del pedido.

setNumeroReferencia(int numeroReferencia)

public void setNumeroReferencia(int numeroReferencia)
Permite establecer el número de referencia del pedido.

Parámetros

numeroReferencia Número de referencia del pedido a establecer.

getIdCliente()

public int getIdCliente()
Permite obtener el id del cliente que realizó el pedido.

devuelve Id del cliente que realizó el pedido.

setIdCliente(int idCliente)

public void setIdCliente(int idCliente)
Permite establecer el id del cliente que realizó el pedido.

Parámetros

idCliente Id del cliente que realizó el pedido.

getIdPez()

public int getIdPez()
Permite obtener el id del pez del que se ha realizado el pedido.

devuelve Id del pez del que se ha realizado el pedido.

setIdPez(int idPez)

public void setIdPez(int idPez)
Permite establecer el id del pez del que se ha realizado el pedido.

Parámetros
idPez Id del pez del que se ha realizado el pedido a establecer.

getPecesSolicitados()

public int getPecesSolicitados()
Permite obtener el número de peces solicitados en el pedido.

devuelve Número de peces solicitados en el pedido.

setPecesSolicitados(int pecesSolicitados)

public void setPecesSolicitados(int pecesSolicitados)
Permite establecer el número de peces solicitados en el pedido.

Parámetros
pecesSolicitados Número de peces solicitados en el pedido a establecer.

getPecesEnviados()

public int getPecesEnviados()
Permite obtener el número de peces enviados.

devuelve Número de peces enviados.

setPecesEnviados(int pecesEnviados)

public void setPecesEnviados(int pecesEnviados)
Permite establecer el número de peces enviados.

Parámetros
pecesEnviados Número de peces enviados a establecer.

DTOPedidoUsuarioPez

Clase pública que representa a un DTO de un pedido que almacena el nombre del pez del que se realiza el pedido, el nombre del usuario que realiza el pedido, el porcentaje de completado del pedido y el número de referencia del pedido.

Atributos

private int numeroReferencia

Número de referencia del pedido.

private String nombreCliente

Nombre del cliente que realiza el pedido.

private String nombrePez

Nombre del pez del que se realiza el pedido.

private double porcentajeCompletado

Porcentaje de completado del pedido.

Interfaces

Interfaz Mar

Interfaz bandera de peces de mar.

Interfaz Río

Interfaz bandera de peces de río.