

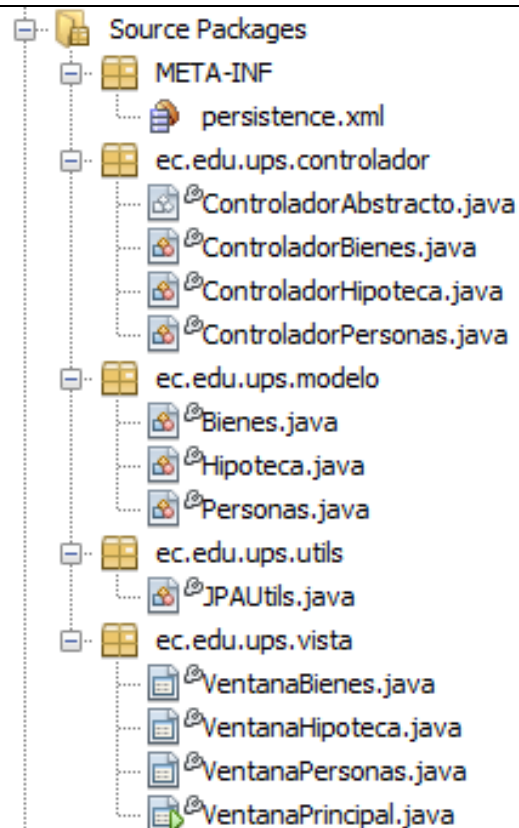


| | | |
|--|-----------------------|---|
|  | Computación | Docente: Diego Quisi Peralta |
| | Programación Aplicada | Período Lectivo: Septiembre 2020 – Febrero 2021 |

| | | | | |
|---|---|--|--|--|
|  | | | FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES | |
| CARRERA: COMPUTACIÓN/INGENIERÍA DE SISTEMAS | | | ASIGNATURA: PROGRAMACIÓN APLICADA | |
| NRO. PROYECTO: | 1 | TÍTULO PROYECTO: Prueba BDD | | |
| OBJETIVO: Analizar conocimientos adquiridos | | | | |
| INSTRUCCIONES: | | Realizar un sistema implementando todos los conceptos vistos en clases para gestionar la hipoteca de las casas con las siguientes características: <ul style="list-style-type: none"> • Las personas compran casas y se convierten en propietarios. • Para pagarlas es habitual que el propietario formalice un préstamo hipotecario con una entidad bancaria. • El banco toma la casa en forma de aval en caso de impago de las mensualidades. • En el caso de que el capital fiado supera el valor de tasación de la casa y el sueldo del propietario no es suficiente, el banco suele exigir la presencia de un avalista (garante). • Para formalizar la hipoteca se necesitan los datos personales del propietario, además de su cédula, dirección de la casa, su dirección, nombres, apellidos y fecha de nacimiento y del garante de ser necesario. • El capital de la hipoteca se ajusta teniendo en cuenta el valor de tasación de la casa y los datos de dirección. • Toda hipoteca se formaliza detallando el capital, el interés (8,99 - 16,99%) y la duración (fecha de inicio y fecha de fin). • A partir de estos datos se calcula el importe de cada mensualidad para el total del tiempo que pide el préstamo. • No es necesario guardar los datos del banco pero si un sistema de autenticación. • Generar los datos con el sistema de amortización Alemán [1]. | | |
| ACTIVIDADES DESARROLLADAS | | | | |



Bienes

```
package ec.edu.ups.modelo;

import java.io.Serializable;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQuery;

/**
 *
 * @author Anahi
 */
@Entity
@NamedQuery(name = "Bienes.findAll", query = "SELECT b FROM Bienes b")
public class Bienes implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    @Column
    private String Direccion;
    @Column
    private String calles;
    @Column
    private String numero;
```

```
@Column
private double precio;

public Bienes() {
}

public Bienes(Long id, String Direccion, String calles, String numero, double
precio) {
    this.id = id;
    this.Direccion = Direccion;
    this.calles = calles;
    this.numero = numero;
    this.precio = precio;
}

public Bienes(String Direccion, String calles, String numero, double precio) {
    this.Direccion = Direccion;
    this.calles = calles;
    this.numero = numero;
    this.precio = precio;
}

public Long getId() {
    return id;
}

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

public String getDireccion() {
    return Direccion;
}

public void setDireccion(String Direccion) {
    this.Direccion = Direccion;
}

public String getNumero() {
    return numero;
}

public void setNumero(String numero) {
    this.numero = numero;
}

public double getPrecio() {
    return precio;
}

public void setPrecio(double precio) {
    this.precio = precio;
}

public String getCalles() {
    return calles;
}

public void setCalles(String calles) {
    this.calles = calles;
}
```

```

    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (id != null ? id.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not
set
        if (!(object instanceof Bienes)) {
            return false;
        }
        Bienes other = (Bienes) object;
        if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "Bienes{" + "id=" + id + ", Direccion=" + Direccion + ", calles=" +
calles + ", numero=" + numero + ", precio=" + precio + '}';
    }
}

```

Hipoteca

```

package ec.edu.ups.modelo;

import java.io.Serializable;
import java.util.Calendar;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.NamedQuery;
import javax.persistence.OneToOne;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

/**
 *
 * @author Anahi
 */
@Entity
@NamedQuery(name = "hipoteca.findAll", query = "SELECT h FROM Hipoteca h")
public class Hipoteca implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

```

```
@OneToOne
@JoinColumn(name="Propietariofk")
private Personas propietario;
@OneToOne
@JoinColumn(name="Avalfk")
private Personas aval;
@OneToOne
@JoinColumn(name="Bienfk")
private Bienes bien;
@Column
@Temporal(TemporalType.DATE)
private Calendar fechaInicio;
@Column
@Temporal(TemporalType.DATE)
private Calendar fechaFinal;
@Column
private double monto;
@Column
private double pagoMensual;
@Column
private double Total;
@Column
private int meses;

public Hipoteca() {
}

public Hipoteca(Long id, Personas propietario, Personas aval, Bienes bien, Calendar
fechaInicio, Calendar fechaFinal, double monto, double pagoMensual, double Total,
int meses) {
    this.id = id;
    this.propietario = propietario;
    this.aval = aval;
    this.bien = bien;
    this.fechaInicio = fechaInicio;
    this.fechaFinal = fechaFinal;
    this.monto = monto;
    this.pagoMensual = pagoMensual;
    this.Total = Total;
    this.meses = meses;
}

public Hipoteca(Personas propietario, Personas aval, Bienes bien, Calendar fe-
chaInicio, Calendar fechaFinal, double monto, double pagoMensual, double Total, int
meses) {
    this.propietario = propietario;
    this.aval = aval;
    this.bien = bien;
    this.fechaInicio = fechaInicio;
    this.fechaFinal = fechaFinal;
    this.monto = monto;
    this.pagoMensual = pagoMensual;
    this.Total = Total;
    this.meses = meses;
}

public Hipoteca(Personas propietario, Bienes bien, Calendar fechaInicio, Calendar
fechaFinal, double monto, double pagoMensual, double Total, int meses) {
    this.propietario = propietario;
    this.bien = bien;
    this.fechaInicio = fechaInicio;
```

```
        this.fechaFinal = fechaFinal;
        this.monto = monto;
        this.pagoMensual = pagoMensual;
        this.Total = Total;
        this.meses = meses;
    }

    public Long getId() {
        return id;
    }

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

    public Calendar getFechaInicio() {
        return fechaInicio;
    }

    public void setFechaInicio(Calendar fechaInicio) {
        this.fechaInicio = fechaInicio;
    }

    public Calendar getFechaFinal() {
        return fechaFinal;
    }

    public void setFechaFinal(Calendar fechaFinal) {
        this.fechaFinal = fechaFinal;
    }

    public int getMeses() {
        return meses;
    }

    public void setMeses(int meses) {
        this.meses = meses;
    }

    public double getMonto() {
        return monto;
    }

    public void setMonto(double monto) {
        this.monto = monto;
    }

    public double getPagoMensual() {
        return pagoMensual;
    }

    public void setPagoMensual(double pagoMensual) {
        this.pagoMensual = pagoMensual;
    }

    public double getTotal() {
        return Total;
    }

    public void setTotal(double Total) {
        this.Total = Total;
    }
}
```

```
}

public Personas getPropietario() {
    return propietario;
}

public void setPropietario(Personas propietario) {
    this.propietario = propietario;
}

public Personas getAval() {
    return aval;
}

public void setAval(Personas aval) {
    this.aval = aval;
}

public Bienes getBien() {
    return bien;
}

public void setBien(Bienes bien) {
    this.bien = bien;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (id != null ? id.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {
    // TODO: Warning - this method won't work in the case the id fields are not
set
    if (!(object instanceof Hipoteca)) {
        return false;
    }
    Hipoteca other = (Hipoteca) object;
    if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
        return false;
    }
    return true;
}

@Override
public String toString() {
    return "ec.edu.ups.modelo.Hipoteca[ id=" + id + " ]";
}
}
```

Personas

```
package ec.edu.ups.modelo;

import java.io.Serializable;
import java.util.Calendar;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQuery;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

/**
 *
 * @author Anahi
 */
@Entity
@NamedQuery(name = "Personas.findAll", query = "SELECT p FROM Personas p")
//@NamedQuery(name = "ConsultaCedula", query = "SELECT p FROM Personas p where
p.cedula = :cedula")
public class Personas implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    @Column
    private String cedula;
    @Column
    private String tipo;
    @Column
    private String nombre;
    @Column
    private String apellido;
    @Column
    private double sueldo;
    @Column
    @Temporal(TemporalType.DATE)
    private Calendar fechaNacimiento;

    public Personas() {
    }

    public Personas(String cedula, String tipo, String nombre, String apellido, double sueldo, Calendar fechaNacimiento) {
        this.cedula = cedula;
        this.tipo = tipo;
        this.nombre = nombre;
        this.apellido = apellido;
        this.sueldo = sueldo;
        this.fechaNacimiento = fechaNacimiento;
    }

    public Personas(Long id, String cedula, String tipo, String nombre, String apellido, double sueldo, Calendar fechaNacimiento) {
        this.id = id;
        this.cedula = cedula;
    }
}
```



```
this.tipo = tipo;
this.nombre = nombre;
this.apellido = apellido;
this.sueldo = sueldo;
this.fechaNacimiento = fechaNacimiento;
}

public Long getId() {
    return id;
}

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

public String getCedula() {
    return cedula;
}

public void setCedula(String cedula) {
    this.cedula = cedula;
}

public String getTipo() {
    return tipo;
}

public void setTipo(String tipo) {
    this.tipo = tipo;
}

public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public String getApellido() {
    return apellido;
}

public void setApellido(String apellido) {
    this.apellido = apellido;
}

public double getSueldo() {
    return sueldo;
}

public void setSueldo(double sueldo) {
    this.sueldo = sueldo;
}

public Calendar getFechaNacimiento() {
    return fechaNacimiento;
}

public void setFechaNacimiento(Calendar fechaNacimiento) {
    this.fechaNacimiento = fechaNacimiento;
}
```

```

    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (id != null ? id.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not
set
        if (!(object instanceof Personas)) {
            return false;
        }
        Personas other = (Personas) object;
        if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "Personas{" + "id=" + id + ", cedula=" + cedula + ", tipo=" + tipo +
", nombre=" + nombre + ", apellido=" + apellido + ", sueldo=" + sueldo + ", fechaNa-
cimiento=" + fechaNacimiento + '}';
    }
}

```

Controlador Abstracto

```

package ec.edu.ups.controlador;

import ec.edu.ups.utils.JPAUtils;
import java.lang.reflect.ParameterizedType;
import java.lang.reflect.Type;
import java.util.ArrayList;
import java.util.List;
import javax.persistence.EntityManager;

/**
 *
 * @author Anahi
 */
public abstract class ControladorAbstracto <E>{
    private List <E> listaGenerica;
    private Class<E> clase;
    private EntityManager em;

    public abstract List<E> findAll();

    public ControladorAbstracto() {
        listaGenerica= new ArrayList();
        //java.lang.reflect.Type t= getClass().getGenericSuperclass();
        Type t = getClass().getGenericSuperclass();
        ParameterizedType pt = (ParameterizedType) t;
        clase= (Class) pt.getActualTypeArguments()[0];
    }
}

```

```
        em=JPAUtils.getEntityManager();
    }
    public ControladorAbstracto(EntityManager em) {
        listaGenerica= new ArrayList();
        //java.lang.reflect.Type t= getClass().getGenericSuperclass();
        Type t =getClass().getGenericSuperclass();
        ParameterizedType pt = (ParameterizedType) t;
        clase= (Class) pt.getActualTypeArguments()[0];
        this.em=em;
    }

    public boolean crear(E objeto){
        em.getTransaction().begin();
        em.persist(objeto);
        em.getTransaction().commit();
        listaGenerica.add(objeto);
        return true;
    }

    public boolean eliminar(E objeto){
        em.getTransaction().begin();
        em.remove(em.merge(objeto));
        em.getTransaction().commit();
        listaGenerica.remove(objeto);
        return true;
    }

    public boolean actualizar(E objeto){
        em.getTransaction().begin();
        em.merge(objeto);
        em.getTransaction().commit();
        return true;
    }

    public E buscar (Object id){
        return(E) em.find(clase, id);
    }

    public List<E> buscarTodo(){
        return em.createQuery("Select t from " + clase.getSimpleName() + "
t").getResultList();
    }

    public List<E> getListaGenerica() {
        return listaGenerica;
    }

    public void setListaGenerica(List<E> listaGenerica) {
        this.listaGenerica = listaGenerica;
    }

    public Class<E> getClass() {
        return clase;
    }

    public void setClass(Class<E> clase) {
        this.clase = clase;
    }

    public EntityManager getEm() {
        return em;
    }
}
```

```

    public void setEm(EntityManager em) {
        this.em = em;
    }
}

```

Controlador Bienes

```

package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Bienes;
import java.util.List;
import javax.persistence.Query;

/**
 *
 * @author Anahi
 */
public class ControladorBienes extends ControladorAbstracto<Bienes>{

    @Override
    public List<Bienes> findAll() {
        Query consulta = getEm().createNamedQuery("Bienes.findAll");
        return consulta.getResultList();
    }

}

```

Controlador Hipoteca

```

package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Hipoteca;
import java.util.List;
import javax.persistence.Query;

/**
 *
 * @author Anahi
 */
public class ControladorHipoteca extends ControladorAbstracto<Hipoteca>{

    @Override
    public List<Hipoteca> findAll() {
        Query consulta = getEm().createNamedQuery("hipoteca.findAll");
        return consulta.getResultList();
    }

}

```


Controlador Personas

```

package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Personas;
import java.util.List;

```

| | | |
|--|-----------------------|---|
|  | Computación | Docente: Diego Quisi Peralta |
| | Programación Aplicada | Período Lectivo: Septiembre 2020 – Febrero 2021 |

```
import javax.persistence.Query;

/**
 *
 * @author Anahi
 */
public class ControladorPersonas extends ControladorAbstracto<Personas> {

    @Override
    public List<Personas> findAll() {
        Query consulta = getEm().createNamedQuery("Personas.findAll");
        return consulta.getResultList();
    }

    public Personas buscarCedula(String cedula){
        Query consulta = getEm().createNamedQuery("ConsultaCedula");
        consulta.setParameter("cedula", cedula);
        return (Personas) consulta.getSingleResult();
    }
}

JPA Utils
package ec.edu.ups.utils;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

/**
 *
 * @author Anahi
 */
public class JPAUtils {
    private static final EntityManagerFactory emf= Persistence.createEntityManagerFactory("PruebaJPAUtils");

    public static EntityManager getEntityManager(){
        return emf.createEntityManager();
    }
}
```

RESULTADO(S) OBTENIDO(S):



File

Gestign Persona

Bienes

Hipoteca

Exit

Crear Persona

Codigo

Cedula

Tipo

Nombre

Apellido

Sueldo

Fecha de Nacimiento

Guardar

Eliminar

| Codigo | Cedula | Tipo | Nombre | Apellido | Sueldo | Fecha Nacimiento |
|--------|--------|------|--------|----------|--------|------------------|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Bienes

Código

Parroquia

Calles

Número

Precio

Guardar

Eliminar

| Código | Dirección | Calles | Número | Precio |
|--------|-----------|--------|--------|--------|
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Propietario

Cédula

Buscar

Aval

Cédula del Aval

Buscar

Tipo

Tipo

Nombre

Nombre

Apellido

Apellido

Sueldo

Sueldo

Fecha de Nacimiento

Fecha de Nacimiento

Bienes

Código

Buscar

Monto

Dirección

Fecha Inicio

Calles

Fecha Final

Número

Meses

Calcular

Precio

Pago total

Hipoteca

Guardar

| Mes | Cuota | Interés | Amortización | Saldo |
|-----|-------|---------|--------------|-------|
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Hipoteca

| Mes | Cuota | Interes | Amortizacion | Saldo |
|-----|---------------|---------------|---------------|---------------|
| 0 | 5065.21739... | 717.391304... | 4347.82608... | 95652.1739... |
| 1 | 5032.60869... | 684.782608... | 4347.82608... | 91304.3478... |
| 2 | 5000.0 | 652.173913... | 4347.82608... | 86956.5217... |
| 3 | 4967.39130... | 619.565217... | 4347.82608... | 82608.6956... |

CONCLUSIONES:

- Los estudiantes implementan soluciones de Bases de Datos JPA

RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la prueba.
- Haber asistido a las sesiones de clase.
- **Consultar con el docente las dudas que puedan surgir al momento de realizar la prueba.**

Docente / Técnico Docente: Ing. Diego Quisi Peralta Msc.

Nombre de estudiante: Edith Anahí Cabrera Bermeo



Firma de estudiante: