Programación Orientada a Objetos. Primer Parcial 1er semestre 2009.
Viernes 17 de abril de 2009

1. Sean las siguientes clases e interfaces, y las siguientes asignaciones:

```
public interface ICelula {
    Decimal DensidadCitoplasmatica {get;set;}
    void Reproducir();
    void AbsorberSustancia();
    void DesecharResiduo();
}

public interface ITermofilo {
    void DisfrutarCalorExtremo();
}

public interface IProcariota: ICelula {}

public interface IEucariota: ICelula {
    Int32 MoverFlagelos();
    Boolean MembranaNuclearIntacta();
}

public interface IEuBacteria: IProcariota {
    void ReplicarPlasmid(){implementacion}
}

public class Aquificales: IEuBacteria, ITermofilo{}

public class Espiroqueta: IEuBacteria {
    public Int32 NumeroEspirales;
}
public class Arqueobacteria: IProcariota, ITermofilo {}

class Program {
    static void Main(string[] args) {
        /*1*/ ICelula      c = new Espiroqueta();
        /*2*/ IEucariota   e = new IEucariota();
        /*3*/ Aquificales  a = new ITermofilo();
        /*4*/ Aquificales  f = c;
        /*5*/ Iprocariota  p = new Espiroqueta();
        /*6*/ IeuBacteria  u = p;
        /*7*/ Itermofilo   t = new Arqueobacteria();
        /*8*/ Iprocariota  r = t;
        /*9*/ IeuBacteria  b = new Aquificales();
    }
  }
```

1.1 Indiquen las líneas de código incorrectas y justifiquen (muy brevemente) su decisión.

Basándote exclusivamente en las líneas correctas del ejercicio anterior, responde las siguientes preguntas:
2.1 ¿Qué tipos tienen los objetos referenciados en las variables definidas en las líneas correctas?
2.2 ¿Qué mensajes puede recibir un objeto como el creado en la línea 5? ¿Cómo lo sabes?
2.3 ¿Qué mensajes puede recibir un objeto bajo los términos de la variable declarada en la línea 7? ¿Cómo lo sabes?

Dadas las siguientes asignaciones:

```
        static void Main(string[] args) {
            Espiroqueta a = new Espiroqueta();
            a.DensidadCitoplasmatica = 5;
            Espiroqueta b = new Espiroqueta();
            b.DensidadCitoplasmatica = 5;
            Aquificales c = new Aquificales();
            c.DensidadCitoplasmatica = 5;
            ICelula e = b;
            e.DensidadCitoplasmatica = 8;
        }
```

3.1 ¿Cuándo dos objetos son iguales?
3.2 ¿Puedes encontrar objetos iguales al final del código anterior? Indícalos.
3.3 ¿Puedes encontrar variables que referencien al mismo objeto? ¿Cuáles?
3.4 Si no encontraste objetos iguales, elimina o agrega las líneas necesarias para obtener por lo menos 2 objetos iguales.

Sea el siguiente fragmento de código:

```
    public interface IFruta { DateTime FechaVencimiento {get;set;} }
    public class Manzana:IFruta     {/*codigo para que compile*/}
    public class Pera:IFruta        {/*codigo para que compile*/}
    public class Carne              { }
    public class Mercado {
          private readonly ArrayList c = new ArrayList();
          public void Agregar(Object o){
             c.Add(o);
          }
          public void Verificar() {
             foreach (Object obj in c){
                  if (obj is Carne) {
                      VerificarEstadoCarnes((Carne)obj);
                  } else{
                      VerificarEstadoFrutas((IFruta)obj);
                  }
             }
          }
          private void VerificarEstadoFrutas(IFruta f) {
             if (f.FechaVencimiento >= DateTime.Today) {
                 Console.WriteLine("Vencido");
             }
          }
          private void VerificarEstadoCarnes(Carne c) {
          }
    }
    class Program {
          static void Main(string[] args) {
              Mercado a = new Mercado();
              a.Agregar(new Manzana());
              a.Agregar (new Pera());
              a.Agregar (new Carne());
              a.Verificar();
          }
    }
```

4.1 ¿Consideras que la operación *`void VerificarEstadoFrutas(IFruta f)`* es polimórfica? Explica brevemente.
4.2 Critíca el código en base al LSP
4.2 Modifica las clases necesarias para que *`Mercado`* sea capaz de verificar el estado de cualquier tipo de comida, y no solo de frutas y carnes.

Sea el siguiente código:

```
public class AlgoMD5 {
	public String CalcularMD5(String text) {
		/* "encripta" complicadamente el texto! */
	}
}
public class AlgoSHA {
	public String CalcularSHA(String text) {
		/* "encripta" super complicadamente el texto! */
	}
}
public class Usuario {
	String email;
	String password;
	/* la password puede almacenarse encriptada por razones de */
	/* seguridad */
	bool encriptada;
	/* el algoritmo usado para guardar la clave */
	String algoritmo;
	bool logueado;
}
public class ServidorEmail {
	/* emails->usuarios */
	private readonly Hashtable usuarios = new Hashtable();
	/* emails->lista de mails */
	private readonly Hashtable mails = new Hashtable();
	/* algoritmos para guardar claves de forma segura */
	private readonly AlgoSHA enc1 = new AlgoSHA();
	private readonly AlgoMD5 enc2 = new AlgoMD5();
	public void Login(String email, String password) {
		Usuario u = usuarios[email];
		u.logueado = False;
		if (u.encriptada) {
			if (u.algoritmo.Equals("SHA1")) {
				password = enc1.CalcularSHA(password);
			} else if (u.algoritmo.Equals("MD5")) {
				password = enc2.CalcularMD5(password);
			}
		}
		if (u.password.Equals(password)) {
			u.logueado = True;
		}
	}
	public String LeerMail(String email) {
		if (usuarios[email].logueado) {
			return mails[email].Remove(0);
		} else {
			return null;
		}
	}
	public void Logout(String email) {
		usuarios[email].logueado = False;
	}
}
```

Justifica si cumple los siguiente principios:

5.1 Encapsulación

5.2 SRP

5.3 Patrón Experto

5.4 OCP

6. Programa los cambios necesarios en el código anterior para que cumpla los principios que no se aplicaban en el código original.

Sea el siguiente fragmento de código:

```
interface ITele {
	public void Prender();
}
public class TeleAlemana {
	public void Anschalten() { /* prende una tele alemana */ }
}
public class ControlRemoto {
	public void Prender(ITele t) { t.Prender(); }
}
public class Program {
	public static void Main(String[] args) {
		ControlRemoto c = new ControlRemoto();
	/* prender una tele alemana con un control remoto!*/
	}
}
```

7. Programa una forma que permita prender una `TeleAlemana` desde `Program` con un `ControlRemoto` **sin modificar**, en nada, la clase `TeleAlemana`.