MODELO EXAMEN PARCIAL N°1
DESARROLLO ORIENTADO A OBJETOS

“Pienso luego existo”.

PARTE TEÓRICA

1) Defina con sus palabras la Abstracción y detalle un ejemplo.

2) Define 2 casos de uso en los que implementaría el método constructor en una clase Usuario.

3) Relacionar de manera lógica y jerárquica donde se podría emplear la herencia y las asociaciones de las siguientes clases (Puede ser de utilidad realizar un diagrama de clases para verlo más claro):

…. Usuario.
…. Administrador.
…. Cliente.
…. Carrito de Compras.
…. Pedido.
…. Detalle de Pedido.

4) Lea el siguiente enunciado y responda:
Se posee la clase MateElectrico y queremos que el atributo float temperatura sea lo más hermético posible, es decir, que solamente mediante los métodos dejarEnfriar() y calentarAgua() definidos por nosotros se pueda disminuir o aumentar el mismo.
¿Que medida debemos tomar para que esto se cumpla?

5) Analizar el siguiente código y buscar los posibles errores en él mismo:

```
class Program
{
    static void Main(string[] args)
    {
        Persona persona = new Persona();
        persona.nombre_completo = "lautaro";
        Usuario[] usuarios = new Usuario[4];

        usuarios[0] = new Usuario("walter nelson", "wallu", "hola123");
        usuarios[1] = new Usuario("felipe pigna", "felipe1", "hola1234");
        usuarios[2] = new Usuario("tommi lopez","friolento", "hola");
        usuarios[3] = new Usuario("cinthia orona", "cin777", "hola777");


        persona.getNombre();
        usuarios[0].getNombre();

        int keyUsuarios = BuscarUsuario(usuarios, "cin777");
        if (keyUsuarios == 1000)
        {
            Console.WriteLine("No existe el usuario");
        } else {
            if ("hola" == (usuarios[keyUsuarios].password)) {
        Console.WriteLine("Bienvenido felipe1 en la posicion "+ keyUsuarios);
```

```
        } else{
                Console.WriteLine("Contraseña incorrecta.");
            }
       }
    }

    public static int BuscarUsuario(Usuario[] usuarios, string nombreUsuario)
    {
        for (int i = 0; i < usuarios.Length; i++)
        {
            if (nombreUsuario.Equals(usuarios[i].nombreUsuario))
            {
                return i;
            }
        }
        return 1000;
    }

}

class Persona
{
    public string nombre_completo;
    public long dni;

    public Persona( string nombre_completo) {
        this.nombre_completo = nombre_completo;
    }

    public virtual void getNombre()
    {
        Console.WriteLine("Mi nombre es " + this.nombre_completo);
    }
}
class Usuario:Persona
{

    public string nombreUsuario;
    public string password;
    public string correo;

    public Usuario(string nombre_completoP, string nombreUsuarioP, string
passwordP)
    {
        this.nombreUsuario = nombreUsuarioP;
        this.nombre_completo = nombre_completoP;
        this.password = passwordP;
    }

    public Usuario(){}

    public void getNombre()
    {
        Console.WriteLine("Mi nombre es " + this.nombre_completo + " y mi
usuario es "+ this.nombreUsuario);

    }
}
```

PARTE PRÁCTICA

Es importante tomarse el tiempo posible para reflexionar respecto a lo que se requiere, segmentar el problema en problemas pequeños y resolver paso a paso.
Se puede recurrir al diagrama de clases si asi lo desea, para verlo en una primera instancia de manera gráfica y fácil de comprender.

1) Se necesita desarrollar una aplicación por consola que nos ayude a almacenar y administrar la información de los integrantes de las células de trabajo de un equipo de desarrollo. Hay dos roles principales: desarrolladores y testers.
Se necesita registrar su **nombre, apellido y edad.**
Además para los desarrolladores se requiere almacenar el **seniority** y el **lenguaje de programación.** Para los testers se debe registrar la **cantidad de proyectos** asignados y el **tipo de test** que realiza (manual o automático).
Se requiere: Definir las clases necesarias para llevar a cabo la administración de los desarrolladores y testers. Incluir los métodos constructores de cada clase y definir un método que devuelva el valor de cada integrante.
El siguiente programa debería funcionar correctamente con las clases definidas.

```
static void Main(string[] args)
{
        var dev1 = new Desarrollador("Tommi", "Perez", 21, “Senior”, "Java");
        var dev2 = new Desarrollador("Cinthia", "Sanchez", 19, “Senior”, "Angular");
        var test1 = new Tester("Walter", "Lopez", 25, 7, “Manual”);
        var test2 = new Tester("Ezequiel", "Gomez", 25, 8,"Automatico");

}
```

¡ÉXITOS!