

Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

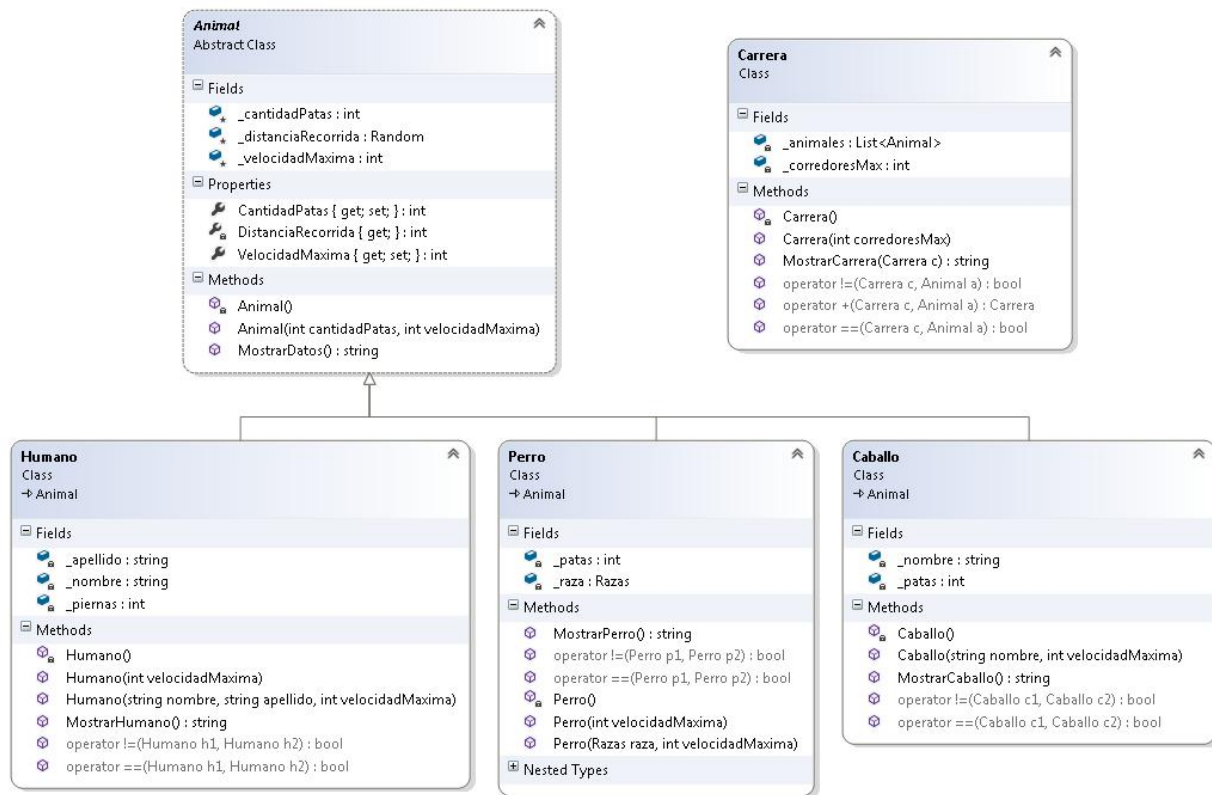
**Materia: LABORATORIO DE PROGRAMACIÓN II**

Apellido:					Fecha:					
Nombre:					Docente <sup>(2)</sup> :					
División:					Nota <sup>(2)</sup> :					
Legajo:					Firma <sup>(2)</sup> :					
Instancia <sup>(1)</sup> :	PP		RPP	X	SP		RSP		FIN	

Generar una Solución nombrada como: *Apellido.Nombre.Division*.

Dentro de un proyecto llamado Entidades, colocar el siguiente esquema de clases:

1. Clase Animal – abstracta:
  - a. Todos los atributos serán Protected.
  - b. El atributo `_distanciaRecorrida` será estático, y sólo se podrá inicializar en un constructor también estático.
  - c. En la propiedad `CantidadPatas` se validará que un animal no pueda tener más de 4 patas, caso contrario se le asignará el número máximo válido.
  - d. En la propiedad `VelocidadMáxima` se validará que un animal no pueda tener una mayor a 60, caso contrario se le asignará el número máximo válido.
  - e. La propiedad `DistanciaRecorrida` de sólo lectura entregará un número aleatorio entre 10 y la velocidad máxima del animal.
2. Clase Humano:
  - a. `_piernas` será estático, y sólo se podrá inicializar en un constructor también estático. El valor será siempre 2.
  - b. Dos humanos serán iguales si su nombre y apellido son iguales.
3. Clase Perro:
  - a. `_patas` será estático, y sólo se podrá inicializar en un constructor también estático. El valor será siempre 4.
  - b. Dos perros serán iguales si tienen la misma raza y la misma velocidad máxima.
  - c. Las Razas serán Galgo y OvejeroAleman.
4. Clase Caballo:
  - a. `_patas` será estático, y sólo se podrá inicializar en un constructor también estático. El valor será siempre 4.
  - b. Dos caballos serán iguales si tienen el mismo nombre.
5. Clase Carrera:
  - a. La lista de animales sólo se podrá instanciar en el constructor privado.
  - b. Una carrera será igual a un animal si contiene un animal del mismo tipo y este cumple con las condiciones de igualdad de la clase correspondiente. ¡**IMPORTANTE!** En este punto, se deberá utilizar un `foreach` y no un `for`.
  - c. Se podrán agregar animales mediante el operador `+` siempre y cuando el animal no figure ya en la lista de la carrera y se tenga cupo según el atributo `_corredoresMax`.
  - d. `MostrarCarrera` expondrá todos los datos de la carrera y de sus competidores.  
¡**IMPORTANTE!** En este punto, se deberá utilizar un `for` y no un `foreach`.



Generar un proyecto de Formularios con el nombre *Apellido.Nombre.Division*:


El formulario tendrá que presentar la siguiente disposición gráfica.

The screenshot shows a Windows Forms application window titled "RPP". Inside the window, there is a form with a light gray background. At the top, there is a button labeled "Prueba Clases". Below it, there is another button labeled "Mostrar Salida Por Pantalla". The rest of the form is empty.

El botón btnPrueba con el texto " Prueba Clases " deberá contener el siguiente código, sin modificación alguna:

```
Animal a1 = new Perro(Perro.Razas.Galgo, 60);
carrera += a1;
Perro a2 = new Perro(Perro.Razas.Galgo, 60);
carrera += a2;
Humano a3 = new Humano("Juan", "Gomez", 20);
carrera += a3;
Caballo a4 = new Caballo("Veloz", 70);
carrera += a4;
Caballo a5 = new Caballo("Match 5", 75);
carrera += a5;
Animal a6 = new Humano("Pedro", "Martínez", 40);
carrera += a6;
Perro a7 = new Perro(Perro.Razas.OvejeroAleman, 50);
carrera += a7;
```

El botón btnMostrar con el texto " Mostrar Salida Por Pantalla " deberá utilizar la Carrera generada en btnPrueba para mostrar los datos de la misma en el RichTextBox llamado rtbSalida.

Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP y dejar este último en el Escritorio de la máquina. Luego presionar el botón  de la barra superior.