Objetivo. Demostrar el avance en el desarrollo de competencias para resolver problemas sencillos utilizando la programación orientada a objetos.

Temas. Diseño de clases. Herencia. Polimorfismo. Sobrecarga.

Instrucciones generales. **Lee** cuidadosamente la descripción del examen y **responde específicamente** lo que se te pide. Calificaré que tu código implemente y **cumpla** con **cada especificación** solicitada, así que te recomiendo mucha atención. Si el examen tiene **errores** de **compilación**, en automático, la calificación máxima a la que aspira es de **setenta puntos**, sin importar el tipo de error.

El examen debe **colocarse** en la plataforma **Canvas**. Este es el único medio de entrega válido para que el examen ser calificado. **Asegura** que subes a la plataforma el **código correcto**, ya que no recibiré ningún código posterior a la entrega del examen.

Entregable. El entregable del examen debe ser **un solo archivo comprimido** que contenga:

- **Todos** los **archivos** requeridos para que el programa funcione.
- Este **documento** (en formato .pdf) donde te pido incluir impresiones de pantalla que muestre el resultado de la ejecución. Puedes agregar las impresiones de pantalla que consideres necesarias. No requiero que des formato a las imágenes.
- Un **comentario** en Canvas que me indique el **IDE** que usaste para construir tu programa.

Nota importante. El examen es **individual** por lo que todo **intercambio** de información causará que el examen sea acreedor de una calificación de **cero puntos**. Original y copia.

****

Especificaciones. Implementa en C++ el código necesario para cumplir con las siguientes especificaciones.

1. Cada **archivo** que implementes debe tener tu **nombre**.
2. **Diseña** una **clase** que modele algún tipo de objetos cuyo nombre inicie con la primera letra de tu apellido materno. Por ejemplo, me apellido materno es Bolaños Cacho, por lo que podría seleccionar implementar la clase Bandera.
3. Declara al menos **2 atributos** en tu clase. Tú decide los tipos de dato de éstos. Recuerda aplicar la propiedad de encapsulamiento.
4. Diseña para tu clase la siguiente **funcionalidad**:
   - Al menos **dos constructores**
   - **Destructor**
   - ***Setters*** y ***getters*** para todos los atributos
   - Sobrecarga el **operador <<** *(muestra todo)*
5. Especializa tu clase con **herencia** en una nueva clase. Decide el nombre.
6. Agrega a esta clase derivada al menos **un atributo**.
7. Agrega a tu clase derivada la siguiente **funcionalidad**:

- Al menos **dos constructores**
- **Destructor**
- ***Setters*** y ***getters*** para todos los atributos
- Sobrecarga del **operador ==** *(tú decide las operaciones, siempre y cuando mantengas la filosofía del operador)*

8. Diseña una aplicación que incluya:
   - Un **arreglo** de tamaño dinámico de objetos. El tamaño se pide al usuario. Valida que este dato sea positivo mayor a cero.
   - Utilizarás objetos de la clase base y de la clase derivada para **llenar** el **arreglo** de la forma en que se explica a continuación.
     - Utiliza el **último dígito de tu matrícula** como referencia (en caso de que este dato sea 0 o 1, utiliza el siguiente dígito hasta que encuentres un dígito que no lo sea). Por ejemplo, mi matrícula es 330839, usaré el 9.
     - Las **posiciones** del arreglo que coincidan con el **dígito** de tu matrícula o dos veces este valor o tres veces este valor, etc., serán llenados con objetos de la clase Base y el resto de los renglones serán llenados con objetos de la clase derivada. Por ejemplo, en mi caso, estas posiciones serían 9, 18, …..
   - **Muestra** el funcionamiento del operador << **desplegando dos objetos** del arreglo.
   - **Muestra** el funcionamiento del operador == comparando los dos objetos que mostraste. Usa un *if* que incluya en cada vía un letrero, por ejemplo, "*son iguales*" y "*son diferentes*"

Coloca en esta sección, **las impresiones de pantalla** que demuestren la ejecución de tu programa.

```
7  //
8  #include "Fútbol.hpp"
9  #include "Cancha.hpp"
10 #include <iostream>
11 using namespace std;
12 int main() {
13     Fútbol sal;
14     sal.getJugadores();
15     sal.getTipo();
16     Cancha salir;
17     salir.getCancha();
18     Fútbol *vec[7];
19     vec[3]=new Fútbol(11,"Fútbol7");
20     vec[3]->getJugadores();
21     vec[6]=new Fútbol(11,"Fútbol7");
22     vec[6]->getTipo();
23     Cancha *ve[10];
24     ve[9]=new Cancha(11,"Fútbol7","rectangular");
25     ve[9]->getCancha();
26
27     return 0;
28 }
29
```

```
jugadores= 6
Tipo= Rápido
cancha: rectangular
jugadores= 11
Tipo= Fútbol7
cancha: rectangular
Program ended with exit code: 0
```