

EVALUACIÓN CORRESPONDIENTE AL PRIMER PARCIAL

FACULTAD:	Tecnología Informática		
CARRERA:	AP		
ALUMNO/A:			
SEDE:		LOCALIZACIÓN:	
ASIGNATURA:	Programación Orientada a Objetos		
CURSO:		TURNO:	
PROFESOR:	Garcia Gustavo	FECHA:	23-5-24
TIEMPO DE RESOLUCIÓN:		EXAMEN PARCIAL NRO:	1
MODALIDAD DE RESOLUCIÓN:	Virtual / Individual		
RESULTADOS DE APRENDIZAJE:			
<div>T4-17-09-1-2-1-RA1: [Identifica] + [las características de los programas orientados a objetos] + [para formular software de calidad] + [utilizando métodos y estrategias estandarizadas]</div> <div>T4-17-09-2-1-2-RA2: [Distingue] + [los fundamentos del paradigma orientado a objetos] + [para formular soluciones de software] + [utilizando las jerarquías y relaciones que propone la OO]</div> <div>T4-17-09-2-3-2-RA3: [Utiliza] + [componentes de software y las tecnologías provistas por el framework] + [para desarrollar software] + [que integre la producción propia y el código reutilizado]</div> <div>T4-17-09-5-1-1-RA4: [Interpreta] + [necesidades funcionales] + [que promuevan el desarrollo de software] + [para mejorar la toma de decisiones en las organizaciones]</div>			

Propósito:

Evaluar la capacidad del estudiante para diseñar y desarrollar un programa orientado a objetos. Su habilidad para administrar el tiempo y los recursos seleccionados para el logro del objetivo propuesto, su capacidad para integrar el marco teórico propuesto con los resultados alcanzados.

CRITERIOS DE EVALUACIÓN:

- Creatividad y originalidad de la propuesta.
- Claridad en la organización de la escritura del código y los comentarios incluidos en el mismo.
- Precisión en el resultado obtenido.
- Utilización de los conceptos abordados en clase
- Racionalidad y coherencia en la validación de los datos ingresados y obtenidos así como en el control de la excepciones

El examen se considerará aprobado con una nota de 4 (cuatro)

Parte Teórica

- 1- ¿Qué son y para que se pueden utilizar las clases anidadas?
- 2- ¿Cómo y para qué se puede aprovechar en la práctica el polimorfismo?
- 3- ¿Para qué se usa una clase abstracta?
- 4- ¿Qué es la sobreescritura?

Parte Practica

Guía de Resolución:

1. Introducción

Desarrollo de un Sistema de Becas para Estudiantes Universitarios

El objetivo de este proyecto es crear un sistema de gestión de becas para estudiantes universitarios. Para ello, se requiere la implementación de dos clases fundamentales:

1. Clase ``Alumno``: Esta clase representará a los estudiantes.
2. Clase ``Beca``: Esta clase representará las becas disponibles.

Adicionalmente, se utilizarán los siguientes componentes para la interacción con el usuario:

- **TextBox**: Para ingresar los datos de los estudiantes y las becas.
- **DataGridView**: Para mostrar la información de los alumnos y las becas de manera tabular.

A continuación, se presentará y desarrollará el código necesario para la implementación de este sistema.

Clase Alumno:

Atributos

Constructor

Clase Beca:

Atributos:

Constructor

Los métodos pueden estar en cualquier clase.

Formulario Principal:

TextBox para ingresar los datos del alumno y becas.

DataGridView para mostrar los alumnos y sus becas.

Botones para agregar alumnos y otorgar becas.

El formulario principal contiene los controles de la interfaz de usuario (TextBox, DataGridView, Botones) y la lógica para interactuar con los objetos Alumno y Beca.

Inicialización de Grilla de Alumnos:

Se crea una grilla (DataGridView) para mostrar los datos de los alumnos. Aquí se configuran las columnas necesarias.

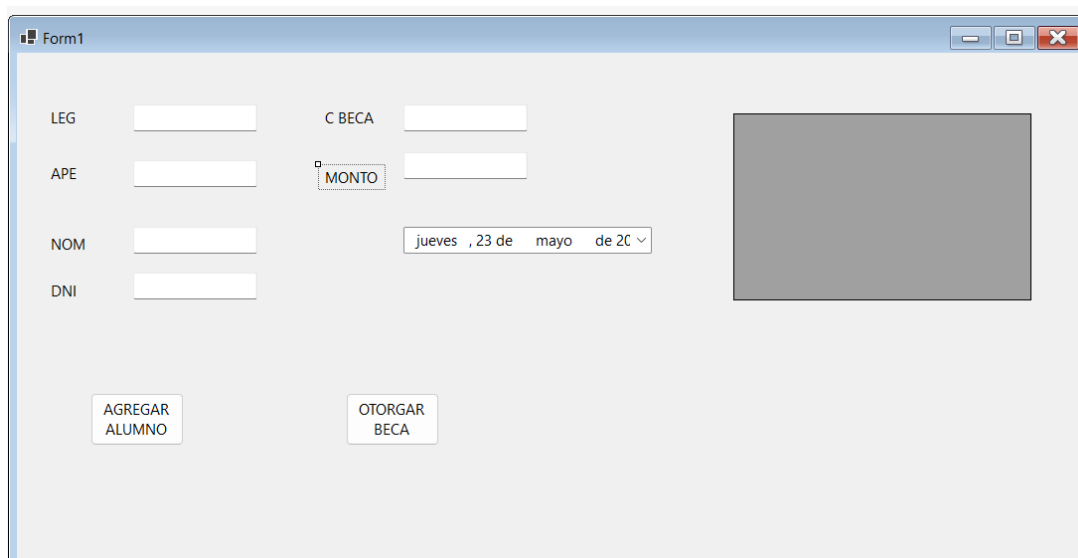
Agregar Alumno:

Al hacer clic en el botón "Agregar Alumno", se lee la información de los TextBox correspondientes y se crea un nuevo objeto Alumno. Luego se agrega este alumno a la lista de alumnos y se muestra en la grilla.

Otorgar Beca:

Cuando se selecciona un alumno de la grilla y se hace clic en el botón "Otorgar Beca", se crea una nueva instancia de Beca y se asigna al alumno seleccionado. Luego se actualiza la grilla para reflejar este cambio.

Ejemplo “este ejemplo es ilustrativo”, no es necesario que sea el mismo a desarrollar.



Utilizar Try ... Catch para administrar las excepciones del sistema.

Observe la usabilidad (fácil de utilizar por el usuario, cantidad de clic para una operación, suma claridad en lo que el usuario debe realizar para utilizar en sistema).

2. Plan de trabajo

Desarrollar el código correspondiente al programa solicitado.

Documentar el código considerando colocar no menos de un comentario cada tres líneas de código.

Probar el código para detectar fallas que no permitan lograr los objetivos planteados.

3. Forma de entrega:

Colocar en un archivo .zip el desarrollo completo y el documento del enunciado del parcial (no utilice ninguna versión que no sea la estándar en .zip ni otras extensiones)

Nombre del Archivo: Asignatura_1erParcial_Apellido_Nombre.zip

Ejemplo: POO_1erParcial_Perez_Juan.zip