

**U**

**NIVERSIDAD**

**A**

**BIERTA**

**I**

**NTERAMERICANA**

**F**

**ACULTAD**

**D**

**E**

**T**

**ECNOLOGÍA**

**I**

**NFORMÁTICA**

**PARCIAL**

**NRO**

**1**

**Asignatura:**

**Pr**

**ogramación I**

**Mesa:**

O

.

Examen

O

.

Recuperatorio de Materia

O

.

Examen Final

**Localización:**

Uai

-

Online

**Turno:**

O

.

Mañana

O

.

Noche

**Fecha**

**:**

**5/10/2020**

**Alumno: Diez Lorenzo Tomas**

**Legajo: 41725136**

**Obs:**

**Nota:**

Requisitos para aprobar: Para que el parcial esté aprobado el alumno deberá tener correctamente desarrolladas el 60% de la teoría y resuelto el ejercicio práctico. Recomendaciones:

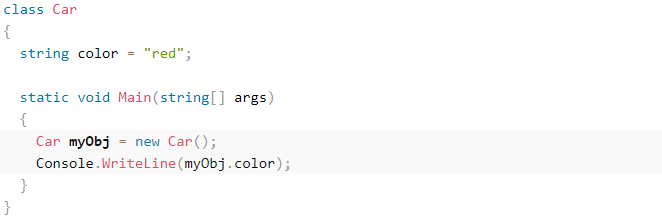
1. Lea todo el parcial antes de comenzar a responder.
2. Desarrolle una redacción clara y precisa contestando lo que la pregunta requiere.
3. Observe la ortografía ya que la misma es parte del parcial.
4. Si considera que no comprende alguna consigna antes de comenzar consulte a su profesor.

# Teoría

1. ¿Qué es una clase? Ejemplifique

Podemos definir a la clase como un plano por medio del cual podemos crear objetos, por ejemplo, un auto. Para crear un auto, pensamos en cómo queremos ese auto y luego diseñamos un plano. El plano no es el auto, pero este nos proporciona las características del auto. El plano sería equivalente a la clase, y el auto el objeto. Una clase nos puede servir para crear muchos objetos independientes pero que tienen las mismas características.

1. ¿Qué es un objeto? Muestre una porción de código de ejemplo relacionado con el punto anterior.

Un objeto es una unidad que tiene un estado y un comportamiento. Es decir, tiene una serie de datos almacenados y tareas que realiza con esos datos en el tiempo de ejecución. Estos objetos se crean instanciando clases. Por ejemplo, volviendo al ejemplo del auto…  


1. ¿Cuál es la diferencia entre un atributo y una propiedad?

Un atributo es una característica individual que diferencia un objeto de otro y determina su apariencia, estado u otra cualidad. Los atributos se guardan en variables denominadas instancia y cada objeto particular puede tener valores distintos. La diferencia principal es que una propiedad es una forma de acceder al contenido de un atributo, tanto para consultar su valor como para modificarlo.

1. ¿Qué es un constructor y para que se utiliza la sobrecarga de constructores?

El constructor es un método especial que podemos utilizar con las clases. Este generalmente es usado para inicializar los valores de los datos con los que trabajará el objeto.  
Su primera característica es que tiene le mismo nombre de la clase y su segunda característica mas importante es que no tiene tipo, es decir, que no solamente no regresa nada, sino que no tiene tipo alguno. El constructor es invocado de forma automática cuando el objeto es instanciado, ya que esto nos da la oportunidad de llevar a cabos cosas en el instante que se instancia el objeto, como, por ejemplo, hacer inicializaciones. Este, puede tener en su interior cualquier código valido de C# y también parámetros o no. Si utilizamos parámetros tendremos que pasar valores necesarios en el momento en el que instanciamos el objeto.

El constructor puede ser sobrecargado, es decir, podemos tener más de una versión del constructor. Esto resulta útil ya que podemos seleccionar como se inicializarán los datos del objeto dependiendo del tipo de constructor que utilicemos. El compilador seleccionara automáticamente el tipo de constructor dependiendo de los tipos y la cantidad de parámetros.

1. ¿Qué es una lista enlazada y qué la diferencia de un vector?

Una Lista Enlazada, consiste en una estructura de datos en la cual los elementos almacenados en la misma pueden ser agregados, borrados y accedidos sin restricciones, en cualquier punto de la estructura. Esta estructura está conformada por secuencia de nodos que puede, en los que se guardan campos de datos arbitrarios y referencias, enlaces o punteros al nodo anterior o posterior. Esto permite ver todos los elementos de la estructura, permitiendo realizar recorridos y consultas de los datos.

La diferencia que posee un vector es que, a diferencia de las estructuras dinámicas, las estructuras estáticas mantienen el tamaño fijo, es decir ya tienen un tamaño predefinido y hacer que estas estructuras cambien de tamaño para poder hacer que el sistema sea más dinámico, requieren un esfuerzo adicional.

1. ¿En qué se diferencian las estructuras de tipo Pila y Cola?

La principal diferencia consiste en su modelo de entrada y salida. Una Pila tiene un modelo LIFO (Last In/First Out), el cual quiere decir que el ultimo elemento que entra, es el primero que sale. Ese elemento es ubicado en el tope de la pila. Sus funciones son: Push (Apilar), Pop (Desapilar), Top/Peek (Cima, tope), Vacía, Llena.

En cambio, una Cola tiene un modelo FIFO (First In/First Out), es decir, primer elemento que entra, es el primer elemento que sale. Los elementos en la que las operaciones de insertar y eliminar se hacen en diferentes extremos. Sus funciones son: Enqueue (Encolar), Dequeue (Desencolar), Peek (Primer elemento de la cola), Vacía, Llena.

1. ¿Por qué es conveniente separar el proyecto de librería, y utilizar solamente las clases públicas como en los ejemplos vistos en clase para las listas, colas y pilas?

Es conveniente ya que, toda la parte de abstracción de nodos, no deberíamos dejar usarla al usuario, no es conveniente que pueda manipularlo por seguridad.

**Página 1 de 2**

# Práctica

Realizar el siguiente ejercicio en código C#:

1. Implementar por consola el uso de una clase “Cola” (Con tamaño máximo establecido por el usuario) que permita ejecutar las siguientes operaciones: Enqueue, Dequeue y Peek, Vacia y LLena. El objeto que podrá “encolarse” será del Tipo Alumno y contendrá los siguientes datos: Nombre: String; Apellido: String; FechaNac: DateTime.

1. Implementar un mecanismo de búsqueda en la propia clase “Cola” que devuelva el primer elemento encontrado a partir del parámetro Apellido. Este mecanismo solo devuelve un objeto de tipo Alumno, no hay que modificar ni eliminar ningún elemento de la Queue.

Para ambos puntos utilizar nodos simplemente enlazados. Realizar casos de prueba desde la consola (Método main()) para validar que funcionen los métodos solicitados y validando los límites establecidos tanto para funcionalidad vacía o llena.

Página 2 de **2**