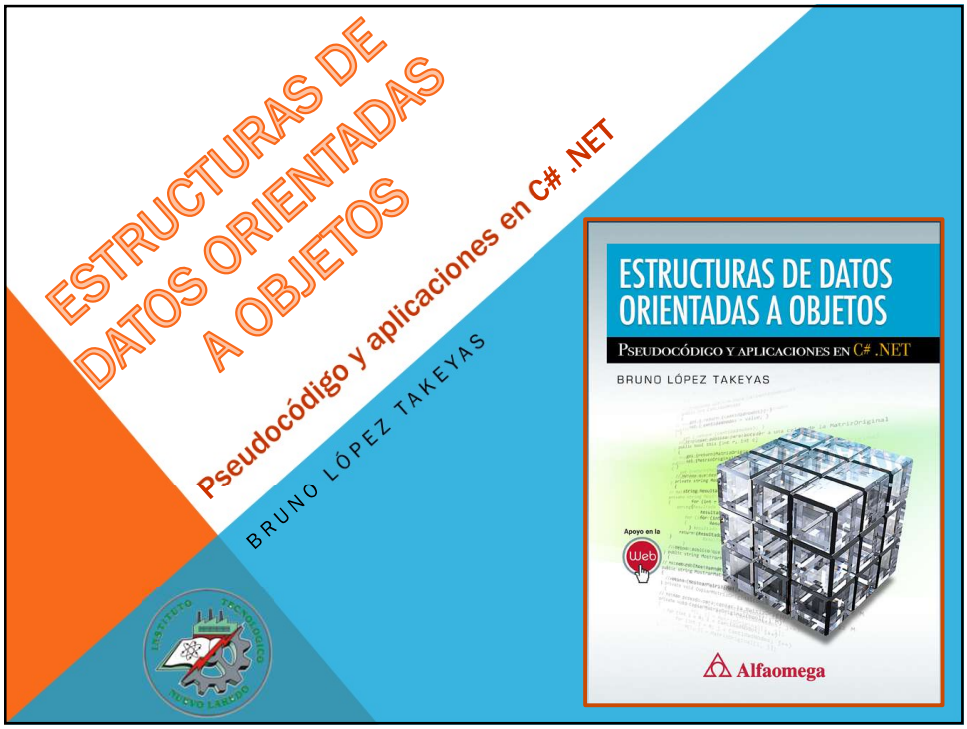
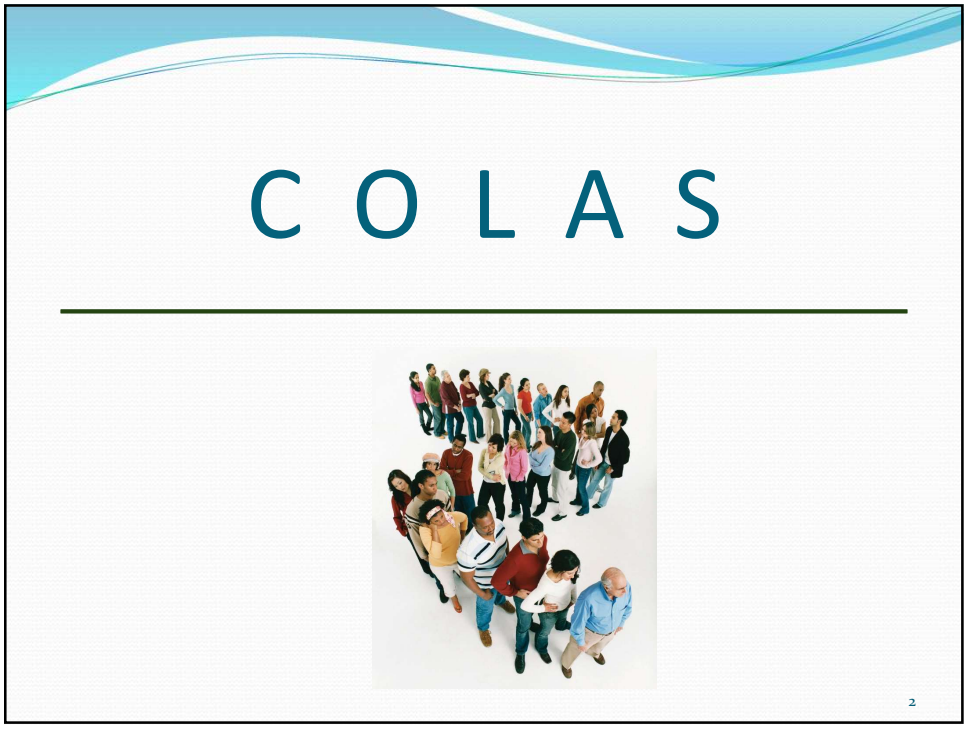

ESTRUCTURAS DE DATOS ORIENTADAS A OBJETOS
Pseudocódigo y aplicaciones en C# .NET
BRUNO LÓPEZ TAKEYAS
ESTRUCTURAS DE DATOS ORIENTADAS A OBJETOS
PSEUDOCÓDIGO Y APLICACIONES EN C# .NET
BRUNO LÓPEZ TAKEYAS
Apoyo en la Web
Alfaomega


C O L A S
2

Preguntas detonadoras

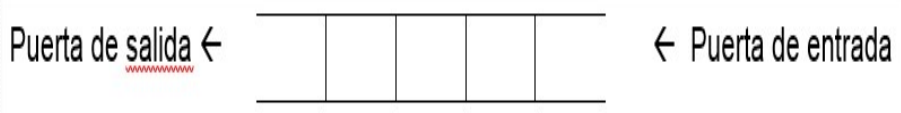
- ¿Qué es una cola?
- ¿Qué características distinguen a una cola?
- ¿Cómo se representa gráficamente una cola?
- ¿Qué operaciones se pueden realizar en una cola?
- ¿Qué situaciones de la vida cotidiana se pueden modelar con colas?
- ¿Existen colas estáticas?
- ¿ … y dinámicas?
- ¿Cómo se diseña un modelo orientado a objetos con colas?

3

Colas

Es una estructura de datos lineal que tiene dos puertas de acceso ubicadas en extremos opuestos, una para insertar los datos y la otra para eliminarlos.

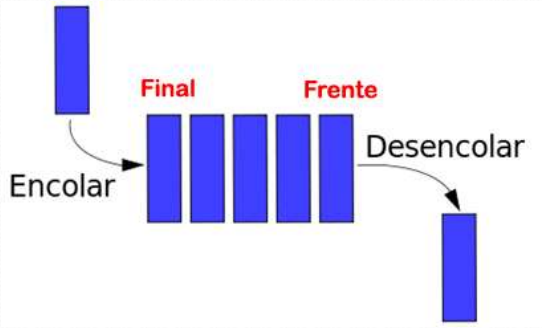
Los datos se insertan uno detrás de otro.



4

Representación gráfica de una cola

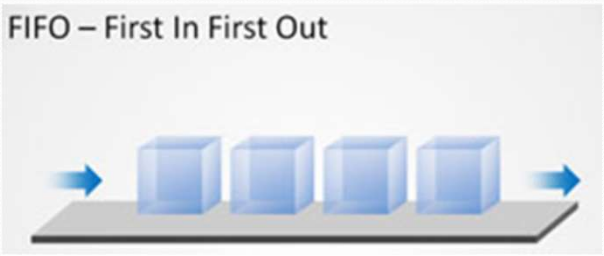
- *La cola tiene dos apuntadores:*
- *Frente.- Puerta de salida de los datos*
- *Final.- Puerta de entrada de los datos*



5

Comportamiento de una cola

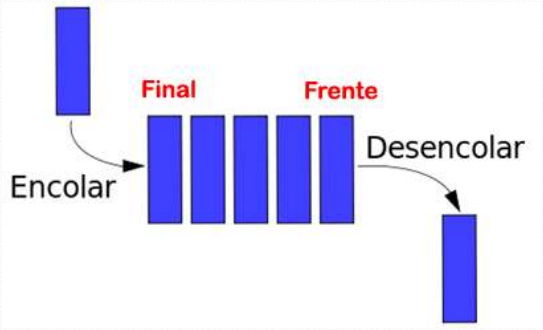
- *FIFO.- First input, first output (primer dato en entrar es el primero en salir)*
- *LILO.- Last input, last output (último dato en entrar es el último en salir)*



6

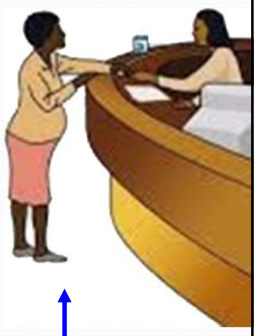
Operaciones básicas con colas

- *Insertar.- Método para insertar un dato en el final de la cola*
- *Eliminar.- Método para eliminar el dato ubicado en el frente de la cola*



7

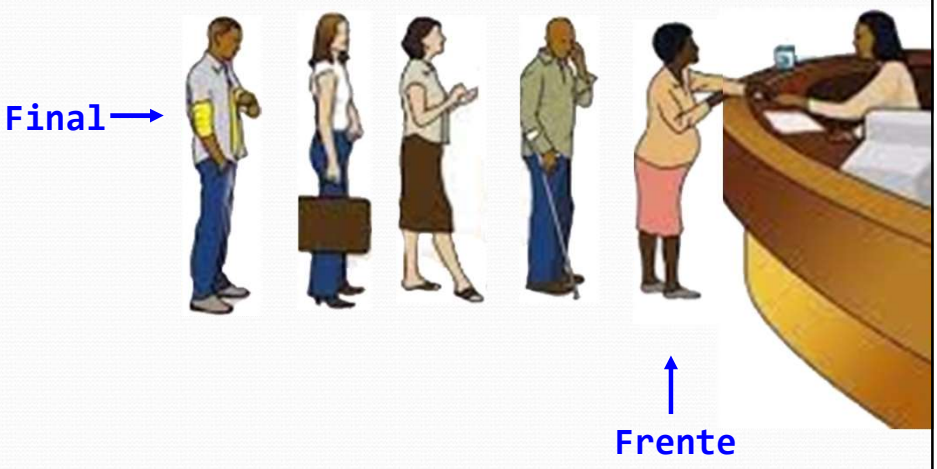
Insertar datos en la cola



Final Frente

8

Eliminar datos de la cola



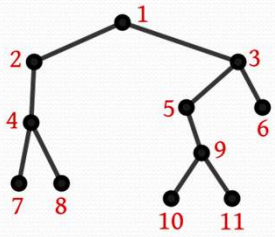
9

Ejemplos de colas en la vida cotidiana

10

Algoritmos que usan colas

Búsqueda en anchura



11

Tipos de colas

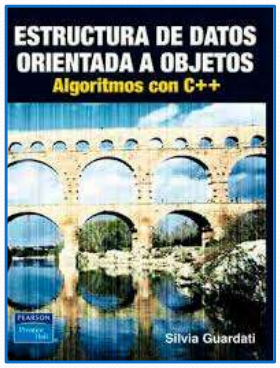
Colas

- *Estáticas.- Basadas en arreglos*
- *Dinámicas.- Basadas en listas enlazadas*

12

Tipos de colas

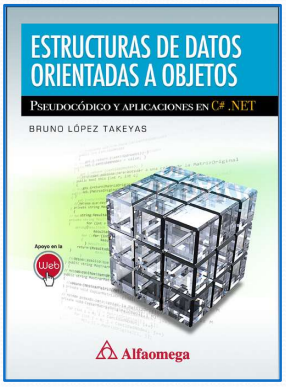
- *Tradicionalmente se imparten las clases de colas de manera estática (basadas en arreglos)*
  - *El libro "Estructura de datos orientada a objetos. Algoritmos con C++" de Silvia Guardati menciona que las pilas y colas pueden representarse de manera dinámica a través de listas.*



Referencias bibliográficas

El libro "Estructuras de datos orientadas a objetos. Pseudocódigo y aplicaciones en C# .NET" aborda las colas de las dos formas:

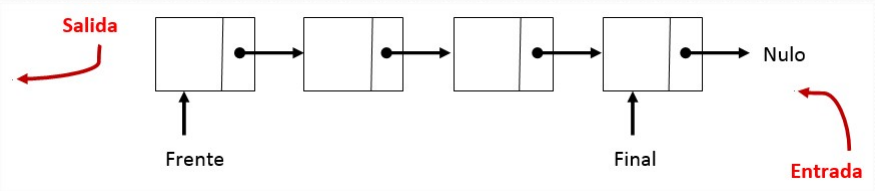
- *Cap. 5.- Colas (estáticas)*
- *Cap.6.- Listas enlazadas (sección 6.10 "Implementación de una cola mediante una lista simple")*



14

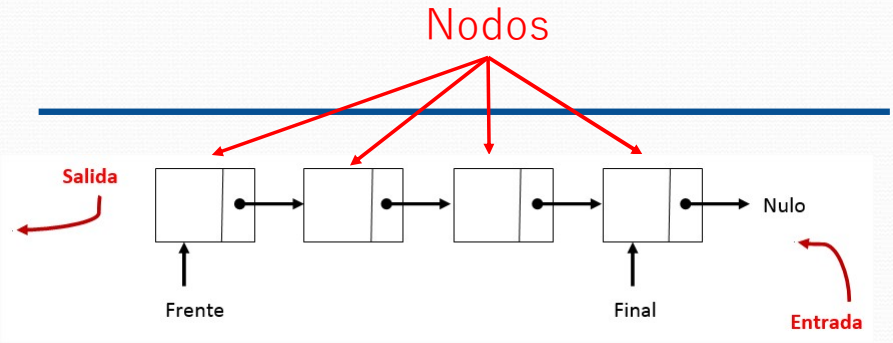
¿Cómo analizaremos las colas?

- *En estos apuntes se analizan las colas dinámicas (basadas en listas) y orientadas a objetos*



15

Representación esquemática de una cola dinámica

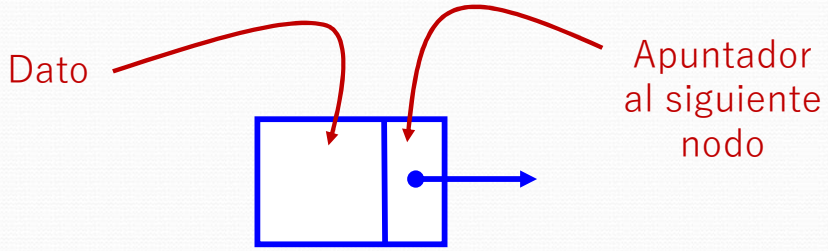


16

Arquitectura de un nodo

Cada nodo tiene 2 secciones:

1. *Dato (puede ser simple o compuesto)*
2. *Apuntador o referencia que enlaza al siguiente nodo en secuencia lógica*



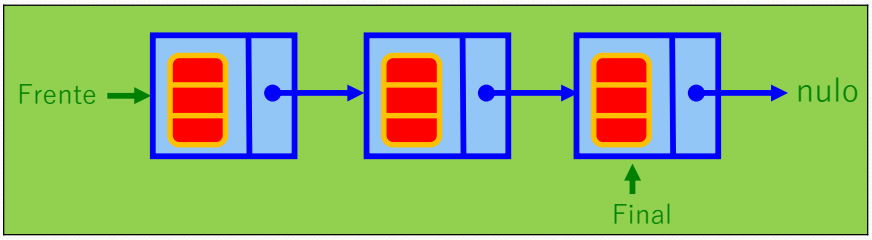
17

Diseño de una cola dinámica orientada a objetos

- Se identifican 3 tipos de objetos en la cola:
  1) *Los objetos con los datos que se desean almacenar en la cola*
  2) *Los objetos de los nodos*
  3) *El objeto de la cola*

18

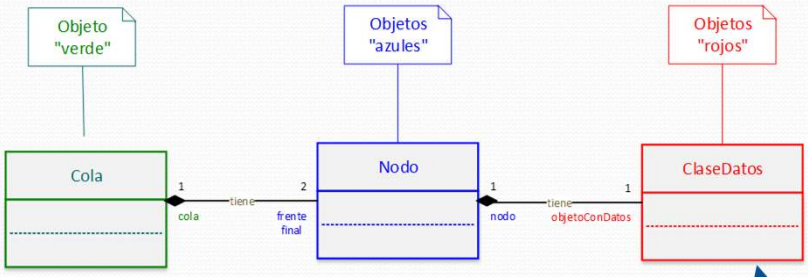
Esquema de los objetos



La **cola (objeto verde)** dentro tiene una secuencia lógica de **nodos (objetos azules)** enlazados por apuntadores y cada uno de ellos a su vez tiene dentro un **objeto con los datos** que se desean almacenar y ordenar **(objetos rojos)**

19

Diseño genérico y didáctico de la cola dinámica orientada a objetos



- *Esta clase puede ser...*
- `Empleado`
- `Estudiante`
- `Médico`
- *etc. (según lo que se desee almacenar en la cola)*

20

Diseño orientado a objetos de una cola dinámica

- Se diseña una cola con objetos creados por medio de varios tipos de clases:
  - *Clase "roja".- Sirve para crear objetos con los datos que se desean almacenar en la cola*
  - *Clase "azul".- Clase para crear los nodos*
  - *Clase "verde".- Clase para crear el objeto de la cola dinámica*

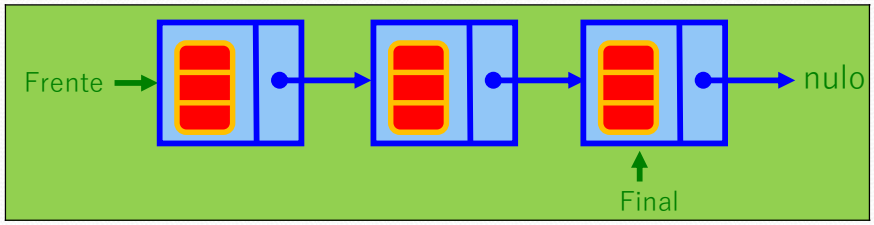
21

Notación de colores

- Los estudiantes deben poner especial atención a los colores usados en las figuras explicativas del resto del curso.
- Los objetos estarán identificados por colores
  - *Objetos "rojos".- Contienen los datos que se desean almacenar en la cola dinámica*
  - *Objetos "azules".- Representan los nodos de la cola*
  - *Objeto "verde".- Es la cola dinámica*

22

Diseño de clases



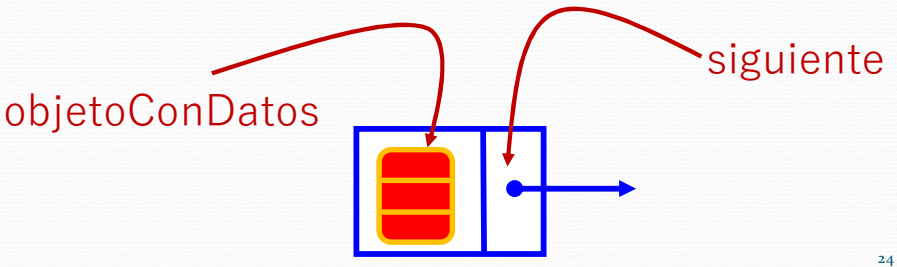
- **Clase "verde"**.- *Define la **composición** entre la cola y el frente y final. También tiene los métodos y propiedades para administrar la cola dinámica.*
- **Clase "azul"**.- *Define los componentes de los nodos.*
- **Clase "roja"**.- *Definiciones de los datos que se desean almacenar en la cola dinámica.*

23

Declaración del nodo

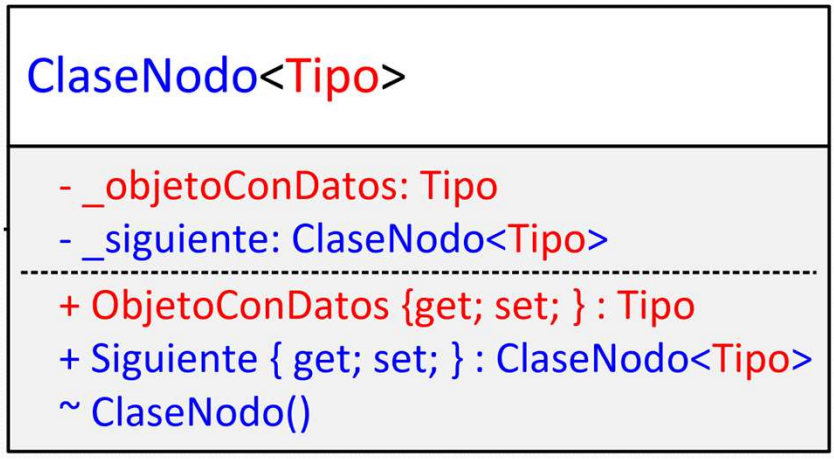
Cada nodo es un obtiene 2 atributos (con sus propiedades):

1. ***objetoConDatos***.- *El nodo recibirá un objeto con los datos que se desean almacenar en la cola*
2. ***siguiente***.- *Apuntador que enlaza al siguiente nodo lógico en la cola*



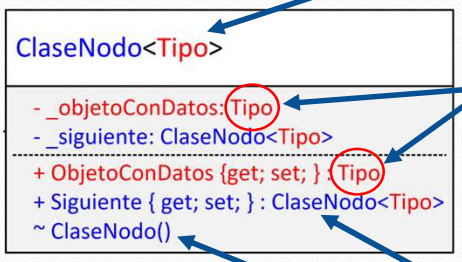
24

Diseño de la clase de los nodos



25

Diseño de la `ClaseNodo`

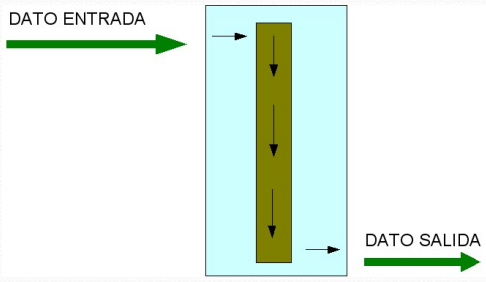


- *Clase parametrizada para recibir cualquier tipo de objeto "rojo"*
- *El parámetro `<Tipo>` define el tipo de objeto "rojo" que estará dentro de cada nodo "azul"*
- *El apuntador `Siguiente` NO almacena otro nodo "azul" sino apunta hacia otro nodo de su mismo tipo*
- *Al eliminar un nodo "azul", su destructor elimina el objeto "rojo" que contiene*

26

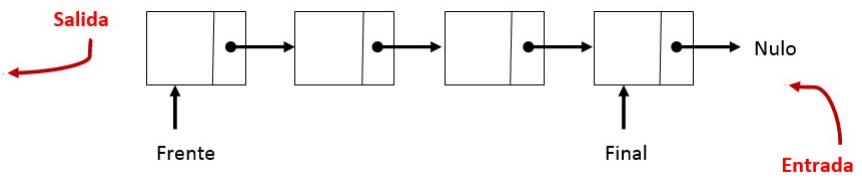
Operaciones en una cola dinámica orientada a objetos

- *Creación de la cola*
- *Insertar*
- *Eliminar( )*
- *Eliminar(dato)*
- *Recorrido*
- *Búsqueda*
- *Eliminar todos los nodos (vaciar)*



27

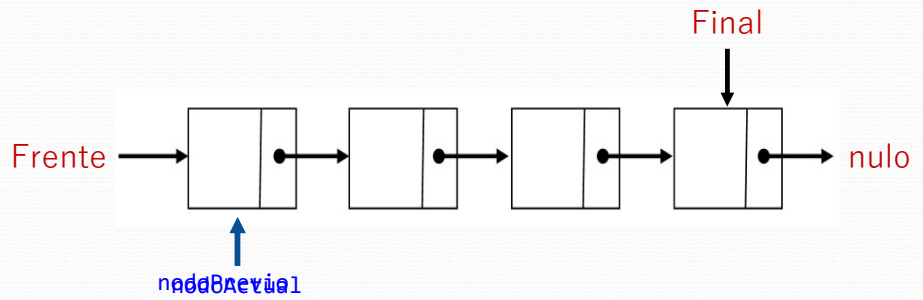
Recorrido de los nodos en una cola dinámica



- *Se verifica que la cola no esté vacía*
- *El recorrido empieza en el* `Frente`
- *Se avanza al próximo nodo a través del apuntador* `Siguiente`
- *En algunos casos es necesario guardar en una variable el nodo previo al cambiar al siguiente nodo*
- *El recorrido termina al llegar al nodo que apunta a nulo o al* `Final`

28

Ejemplo de recorrido en una cola dinámica

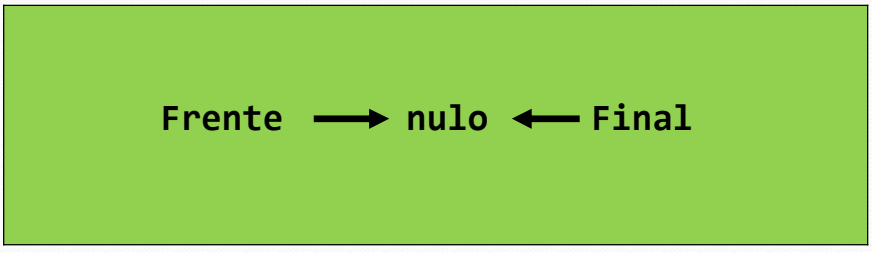


Termina el recorrido

29

Creación de una cola

Cuando se crea una cola tanto el frente como el final el apuntan a nulo



30

Situaciones críticas

- *Son las situaciones que se pueden presentar al realizar operaciones con estructuras de datos*
- *El programador debe prever para diseñar algoritmos eficientes*

31

Método Insertar (Encolar)

- Situaciones críticas:
  - *Alta a cola vacía*
  - *Alta al final.- Se inserta el dato al final de la cola*
  - ***No** permitir datos duplicados*

Se debe recorrer la cola completa antes de insertar un nodo nuevo para validar que no exista duplicado

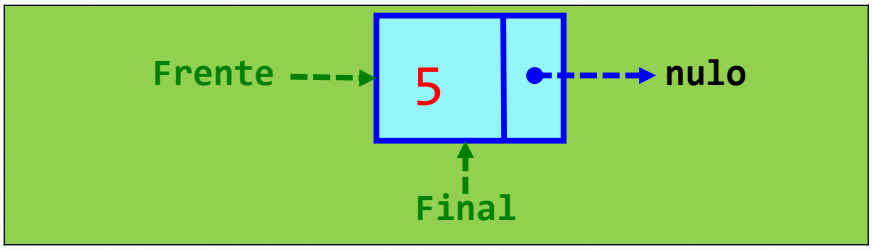
Recuerde:
La cola almacena los datos desordenados

32

Insertar en una cola dinámica vacía

Cuando se detecta la cola dinámica vacía:

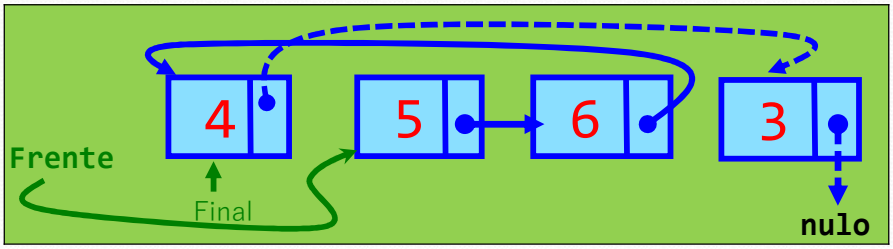
1) *Crear el nodo nuevo "azul"*
2) *Insertar el dato "rojo"*
3) *El nodo nuevo apunta a nulo*
4) *El Frente y el Final apuntan al nodo nuevo*



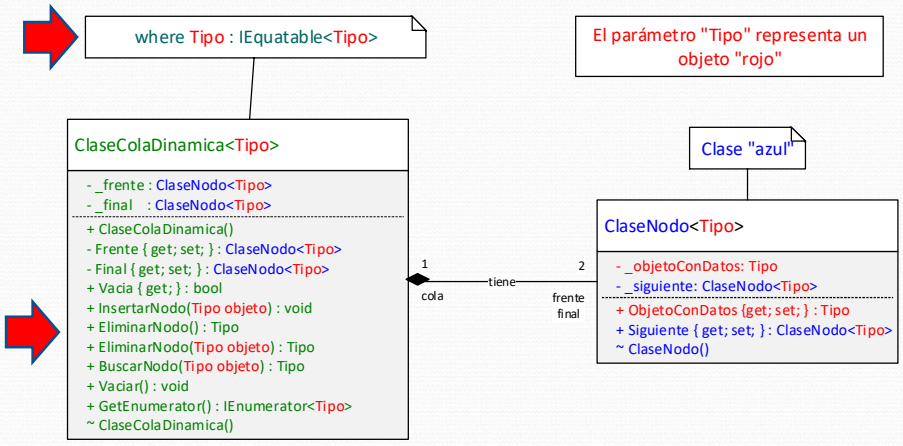
Alta al final en una cola

Cuando se inserta un dato al final de la cola:

1) *Crear el nodo nuevo "azul"*
2) *Insertar el dato "rojo"*
3) *El nodo previo apunta al nodo nuevo*
4) *El nodo nuevo apunta a nulo*
5) *El final apunta al nodo nuevo*



Diseño de la clase de la cola dinámica



35

Composición Cola-Nodo

- *¿Por qué es una composición **1..2** si la cola tiene muchos nodos dentro de ella?*
  - *Porque la cardinalidad de una composición se define por la cantidad de atributos de tipo "parte" contenidos en la clase del "todo" (regla 1 de la composición)*

36

ClaseColaDinamica<Tipo>

- *Clase parametrizada para prepararla para que pueda recibir cualquier `Tipo` de objeto "rojo"*
- *Tiene una restricción de tipos para "obligar" a que la clase "roja" implemente `IEquatable`*
  - *Se requiere el método `Equals()` para buscar un objeto "rojo" almacenado en la cola*
- *¿Por qué no tiene restricción de tipos para `IComparable`?*

37

Componentes de la clase

- `_frente`.- *Atributo privado que apunta al primer nodo de la cola*
- *_final.- Atributo privado que apunta al último nodo de la cola*
- `ClaseColaDinamica()`.- *Es el constructor que inicializa vacía la cola.*
- `Vacia`.- *Propiedad pública booleana de solo lectura para detectar si la cola está vacía (devuelve `true` si la cola está vacía).*

38

Componentes de la clase *(cont.)*

- `Frente`.- *Propiedad privada que apunta al primer nodo de la cola*
- `Final`.- *Propiedad privada que apunta al último nodo de la cola*
- `Insertar(Tipo objeto):void` .- *Método público que recibe como parámetro el objeto "rojo" que se desea almacenar en la cola.*
- `Eliminar():Tipo` .- *Método público que **NO** recibe parámetro y elimina el nodo ubicado en el frente de la cola. **Devuelve** el objeto "rojo" eliminado.*
- `Eliminar(Tipo objeto):Tipo` .- *Método público que recibe como parámetro el objeto "rojo" intermedio o final que se desea borrar de la cola. **Devuelve** el objeto "rojo" eliminado.*

39

Componentes de la clase *(cont.)*

- `BuscarNodo(Tipo objeto):Tipo` .- *Método público que recibe como parámetro el objeto "rojo" que se desea consultar en la cola. Devuelve el objeto "rojo" localizado.*
- `Vaciar():void` .- *Método público que recorre la cola para eliminar todos los nodos "azules" con sus respectivos objetos "rojos"*

40

Componentes de la clase *(cont.)*

- `GetEnumerator():IEnumerator<Tipo>`.- *Iterador que recorre cada nodo "azul" de la cola para consultar su objeto "rojo"*

- `~ClaseColaDinamica()`.- *Destructor que invoca al método `Vaciar()` para eliminar todos los `nodos` de la cola.*

41

Tarea 2.09.- DF Colas dinámicas (constructor, vacía e iteradores)

- Subir a MS Teams los archivos *.vsdx con diagramas de flujo de:
  - *Constructor de la cola dinámica*
  - *Propiedad para detectar si la cola dinámica está vacía*
  - *Iterador GetEnumerator()*

42

Diseño de la clase "roja"

- *Requisitos: Debe contener al menos un dato de los siguientes tipos:*
  - *Int*
  - *Double*
  - *String*
  - *Char*
  - *DateTime*
  - *Bool*
  - *String con la ruta del archivo que contiene una fotografía del objeto*

Se recomienda consultar las filminas
"El lenguaje C# y diseño de formas"
Para usar el PictureBox

43

Diseño de la clase "roja" (*cont.*)

IMPORTANTE

Considere un dato dentro de su clase "roja" que sea único e irrepetible que sirva para identificar a un objeto y que sea el criterio de comparación entre objetos

(Una clave para identificar y comparar objetos "rojos")

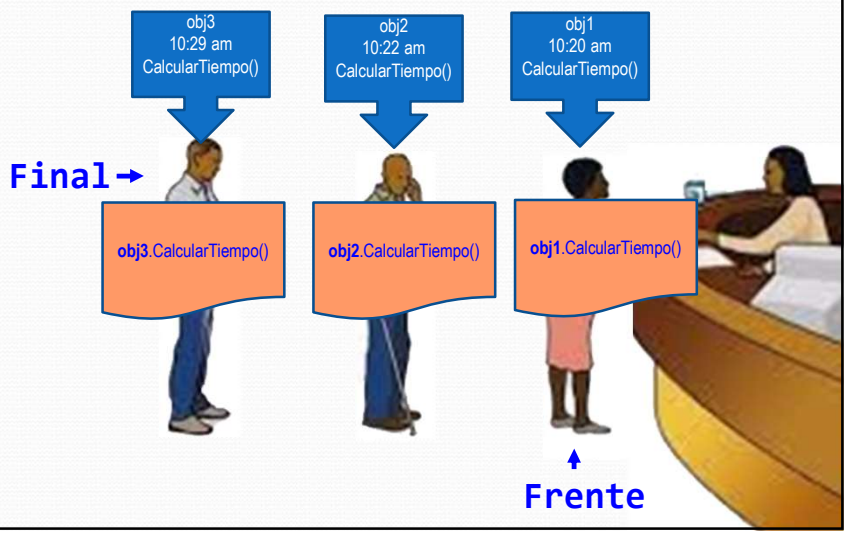
44

Diseño de la clase "roja"

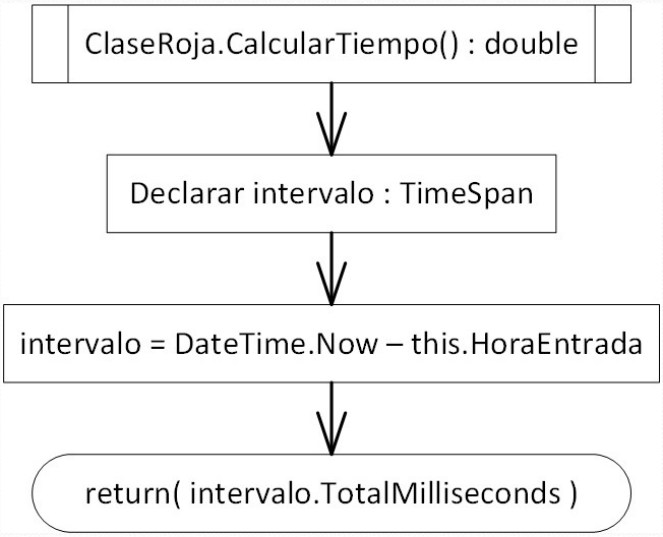
- *Agregar atributo y propiedad para almacenar la fecha y hora de entrada a la cola*
  - `_dtmHoraEntrada: DateTime`
  - `HoraEntrada {get; set; }: DateTime`
- *Agregar un método que calcule el tiempo en la cola*
  - `CalcularTiempo():double`

Se recomienda devolver el tiempo en milisegundos

Calcular el tiempo en la cola



Calcular diferencia de tiempo



47

Tarea 2.10

- *Resolver la Tarea 2.10.- Diseño de la clase "roja" de la cola dinámica en MS Teams*
- *Subir el archivo *.vsdx con el diagrama de la clase "roja" elaborado en Microsoft Visio*
- *Incluir las interfaces correspondientes y método para calcular el tiempo*

48

Tarea 2.11.- DF Colas dinámicas (Agregar)

- Subir a MS Teams el archivo *.vsdx con diagrama de flujo de:
  - *Método para agregar un objeto "rojo" a la cola dinámica*

49

Diseño de la forma de la aplicación visual

- *Requisitos: Debe contener al menos uno de estos controles visuales:*
  - *Textbox*
  - *Button*
  - *ComboBox*
  - *DateTimePicker*
  - *CheckBox*
  - *PictureBox*
  - *DataGridView*
  - *RadioButton*

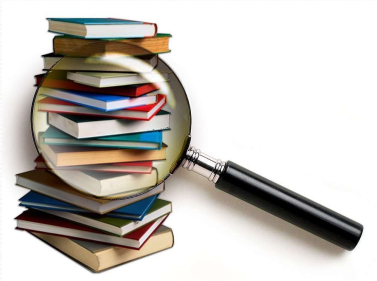
Elija el control adecuado para cada dato capturado

Se recomienda consultar las filminas "Uso de los controles visuales"

50

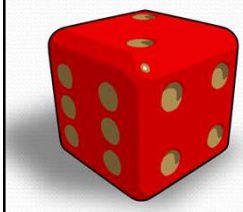
Investigación

- *Agregue un botón a su aplicación para crear `10` nodos con datos generados de manera aleatoria*
- *Muestre los datos generados en el dataGridView*



51

LECTURA



Para generar datos aleatorios, se recomienda la lectura de los apuntes

¿Cómo generar datos aleatorios en C#?

Incluye generar:

- *Nombres aleatorios*
- *Sexo*
- *Fecha*
- *Datos booleanos*
- *etc.*

https://nlaredo.tecnm.mx/takeyas/Apuntes/Fundamentos%20de%20Programacion/Apuntes/07.-Aleatorios.pdf

52

Tarea 2.12.- Diseño de la forma de colas dinámicas

- Diseñar la forma en C#:
- *Capturar los datos usando los controles visuales adecuados*
- *Agregar un dataGridView de solo lectura para visualizar los datos de los objetos "rojos"*

- *Implementar el método para agregar objetos "rojos" a la cola dinámica y visualizarlos en el dataGridView*
- *Subir a MS Teams un archivo comprimido con la aplicación completa (P. ejem. LopezTakeyasBruno.ZIP)*

53

Tarea 2.13.- DF Colas dinámicas (Buscar)

- Subir a MS Teams el archivo *.vsdx con diagrama de flujo de:
  - *Método para buscar un objeto "rojo" en la cola dinámica (debe devolver el objeto encontrado)*

54

Eliminación de datos en una cola dinámica orientada a objetos

- *Se sobrecarga el método* `Eliminar()`
- `Eliminar()`.- *No recibe parámetros porque simplemente elimina el nodo apuntado por el* `Frente`
- `Eliminar(Tipo objeto)`.- *Recibe como parámetro el objeto "rojo" intermedio o al final que se desea eliminar de la cola*
  - *Durante el recorrido es necesario "guardar" el nodo previo*
  - *No se puede interrumpir la búsqueda por anticipado ya que la cola almacena datos desordenados*

55

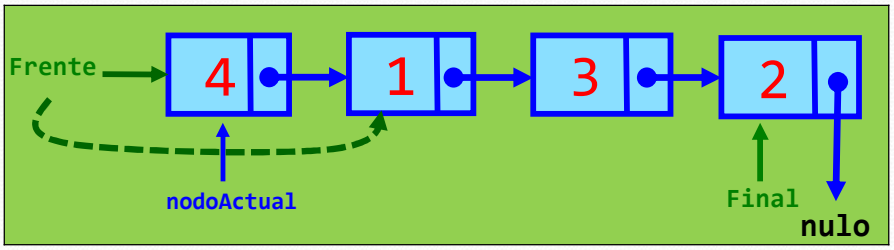
Situaciones críticas del método Eliminar()

- *Si la cola está vacía.- Se dispara una excepción*
- *Baja al principio.- Se elimina el dato apuntado por el* `Frente`
- *Baja del único nodo.- Se elimina el único dato y el* `Frente` *y el* `Final` *quedan apuntando a nulo*

Método Eliminar( )

Cuando se elimina el nodo ubicado en el frente de la cola:

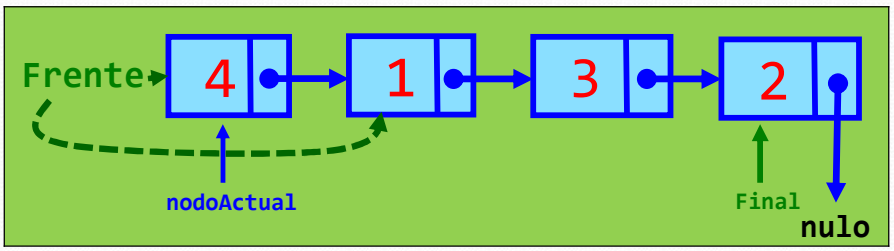
1) *El Frente apunta al nodo que apuntaba el primer nodo de la cola*
2) *Se elimina el nodoActual*



Método Eliminar(Tipo objeto)

Cuando se elimina el nodo ubicado en el frente de la cola:

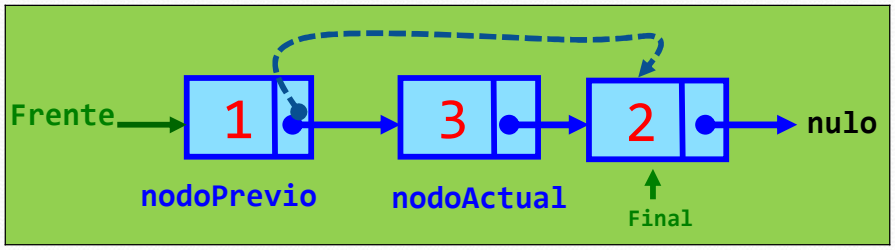
1) *El Frente apunta al nodo que apuntaba el primer nodo de la cola*
2) *Se elimina el nodoActual*



Eliminar(Tipo objeto).- Baja intermedia en una cola

Cuando se elimina un nodo intermedio:

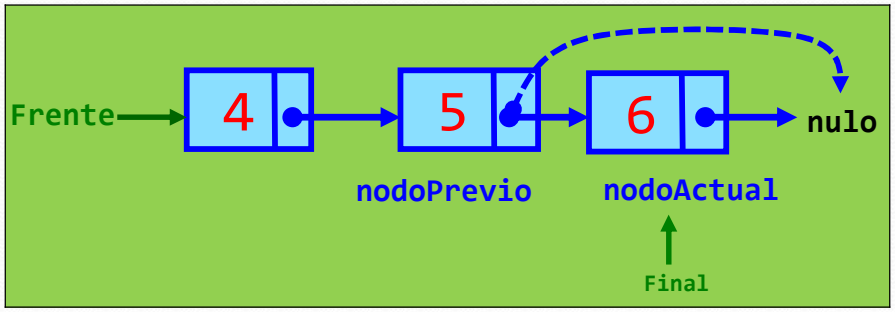
1) *El nodoPrevio apunta al nodo que apuntaba el nodoActual*
2) *Se elimina el nodoActual*



Eliminar(Tipo objeto).- Baja al final en una cola

Cuando se elimina el nodo ubicado al final de la cola:

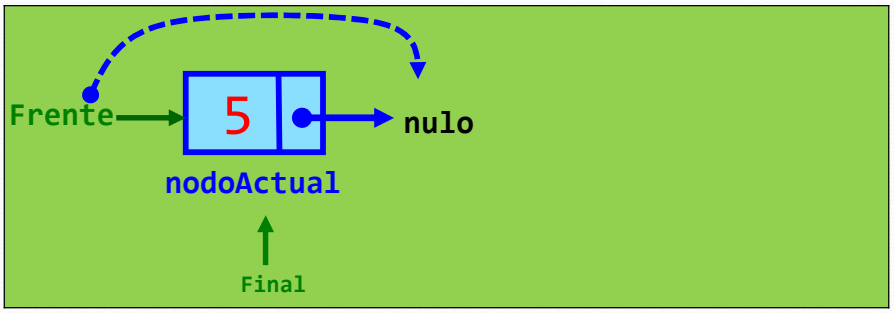
1) *El nodoPrevio apunta al nodo que apuntaba el nodoActual (ahora apunta a nulo)*
2) *Se elimina el nodoActual*
3) *El final apunta al nodoPrevio*



Eliminar el único nodo de una cola

Cuando se elimina el único nodo de la cola:

1) *El frente y el final apuntan a nulo*
2) *Se elimina el nodoActual*



Eliminación de objetos

- *La forma natural de borrar un objeto es asignarle el valor null*

```
// Creación del objeto "azul" del nodoActual
ClaseNodo<Tipo> nodoActual = new ClaseNodo<Tipo>();
   .
   .
nodoActual = null; // Eliminación del nodoActual
```

- *Sin embargo, no todos los datos aceptan el valor null, entonces… ¿cómo se eliminarían?*

62

Destructor de la clase "azul"

- *La `ClaseNodo<Tipo>` es parametrizada y está preparada para recibir un objeto "rojo" de cualquier tipo.*
- *El destructor de la clase "azul" elimina el objeto con datos "rojo" que contiene.*
- *Utiliza `default(Tipo)` para eliminar el objeto "rojo" porque desconoce si el ObjetoConDatos acepta el valor null.*

```
~ClaseNodo() // Destructor de la clase "azul"
{
     // Elimina el ObjetoConDatos "rojo"
     ObjetoConDatos = default(Tipo);
}
```

Tarea 2.14.- DF Colas dinámicas (Eliminar)

- Hacer el diagrama de flujo de:
  - Método `Eliminar( )`
    - *Agregar un botón "Eliminar frente" para este método*
  - Método `Eliminar(Tipo objeto)`
    - *Seleccionar un renglón del dataGridView y después oprimir el botón "Eliminar"*
    - *Al seleccionar un renglón del dataGridView se deben mostrar los datos en los controles adecuados*
  - En ambos casos:
    - *Confirmar la operación*
    - *Subir los diagramas en archivos separados*
    - ***Devolver el objeto "rojo" eliminado***

64

Vaciar la cola

- Situaciones críticas:
  - *Verificar si la cola ya está vacía*
  - *Recorrer los nodos "azules" de la cola*
  - *Durante el recorrido, guardar el* `nodoPrevio`
  - *Eliminar el* `nodoPrevio`
  - *Al terminar, el* `Frente` *y el* `Final` *deben apuntar a nulo*

65

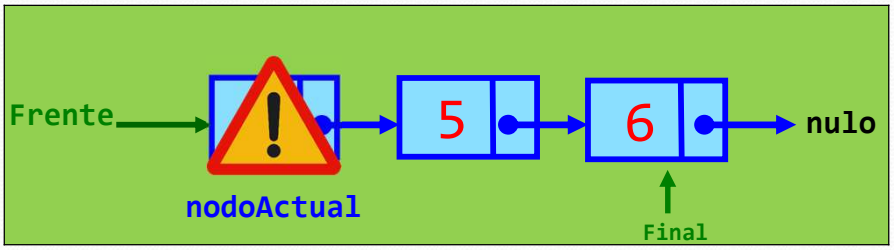
Precaución al vaciar la cola

- **NOTA IMPORTANTE:**
  - *Durante el recorrido de los nodos "azules" de la cola,* ***NO*** *se debe eliminar el* `nodoActual`
  - *Si se eliminara el* `nodoActual` *se pierde el apuntador* `Siguiente`
  - *Se perdería la secuencia lógica de los nodos "azules"*

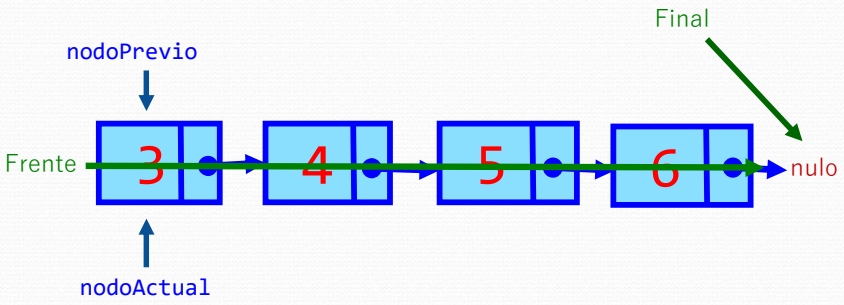
66

Precaución al vaciar la cola

- NOTA IMPORTANTE:
  - *Si se eliminara el `nodoActual` . . .*
  - *¿Cómo verificaría cuál es el siguiente nodo?*
  - *Por lo tanto **NO** debe eliminarse el `nodoActual` sino el `nodoPrevio`*



Ejemplo: Vaciar la cola



Termina el recorrido

68

Tarea 2.15.- DF Colas dinámicas (Vaciar y Destructor)

- Subir a MS Teams los archivos *.vsdx con diagramas de flujo de:
  - *Método para vaciar la cola dinámica*
  - *Destructor de la cola dinámica*

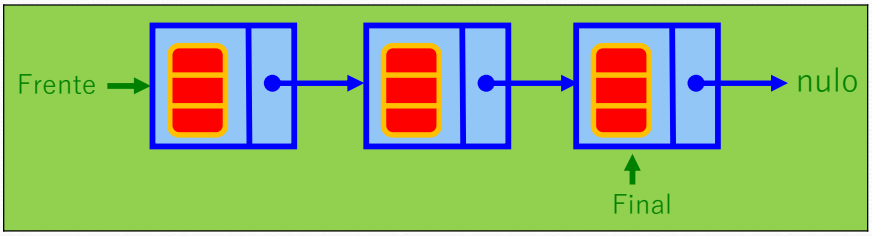
69

Tarea 2.16.- Aplicación completa de colas dinámicas

- *Agregar un botón "Vaciar"*
- *Solicitar al usuario que confirme las operaciones (insertar, eliminar, vaciar, etc.). Preguntarle si está seguro que desea realizar la operación solicitada*
- *Mostrar los mensajes adecuados*
- *Subir a MS Teams un archivo comprimido con la aplicación completa (P. ejem. LopezTakeyasBruno.ZIP)*

70

Nivel de abstracción



- *No se debe perder de vista que los objetos "rojos" son privados, por lo tanto sus componentes son inacesibles para el objeto "verde"*

71

Nivel de abstracción (*cont.*)

- *El objeto "verde" recibe un objeto "rojo" y lo compara para almacenarlo ....*

  ¡¡¡ SIN SABER LO QUE TIENE DENTRO !!!

- *¿Cómo es posible que el objeto "verde" compare y almacene objetos "rojos" sin tener acceso a sus componentes?*

72

"Pensar en objetos …"

- *El objeto "verde" **NO** compara los objetos "rojos" (ellos mismos se comparan entre sí).*
- *El objeto "verde" **NO** requiere acceso a los componentes de los objetos "rojos" para manipularlos.*
- *Recibe cualquier tipo de objetos "rojos" …*

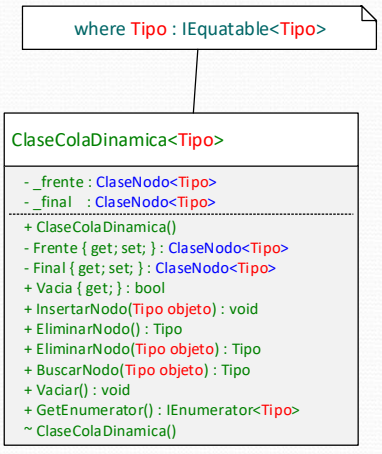
¡¡¡ SIN MODIFICAR NI UNA LÍNEA DE SU CÓDIGO!!!

- ***¿Cómo lo logra? …***
  - *Clases parametrizadas*
  - *Uso de interfaces*
  - *Composición*
  - *Comportamiento polimórfico*
  - *Restricción de tipos*

73

Diseño de una clase polimórfica

- La clase de la cola tiene estas características:
  - *Almacena objetos "rojos" desordenados*
  - *No permite duplicados*



74

Diseño de una clase polimórfica *(cont.)*

- *¿Qué cambios se harían para que la misma clase "**verde**" almacene los datos ordenados?*
- *¿Qué cambios se harían para que la misma clase "**verde**" permita objetos "rojos" duplicados?*

75

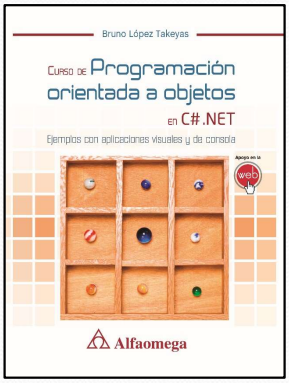
Diseño de una clase polimórfica *(cont.)*

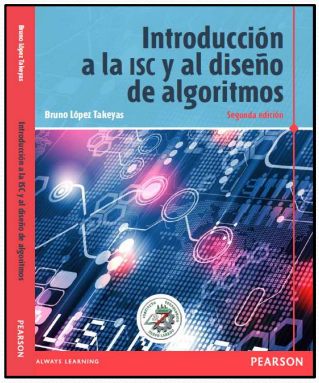
- *¿Qué cambios se harían para que la misma clase "**verde**" muestre cualquiera de los siguientes comportamientos:*
  - *Objetos "rojos" ordenados sin duplicados*
  - *Objetos "rojos" ordenados con duplicados*
  - *Objetos "rojos" desordenados sin duplicados*
  - *Objetos "rojos" desordenados con duplicados?*

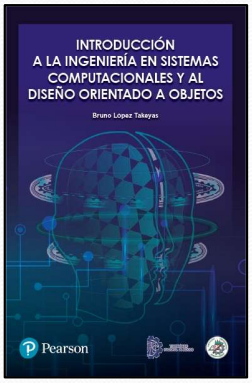
76

Otros libros del autor

https://nlaredo.tecnm.mx/takeyas/Libro







bruno.lt@nlaredo.tecnm.mx

Bruno López Takeyas