Guía de Estudio: Clase 03

**Objetos:**

1. ¿Los objetos se crean en tiempo de diseño, de compilación o deejecución?

Se crean en ***tiempo de ejecución***.

Se denomina **tiempo de ejecución** (*runtime* en inglés) al intervalo de tiempo en el que un programa de computadora se ejecuta en un sistema operativo.

2. ¿Cuántos objetos de tipo Alumno puedo tener en mi sistema? ¿Cuántas clases Alumno tengo en mi sistema? (Dentro de un mismo namespace).

Se pueden tener tantos objetos Alumno como se desee, y dentro de un mismo namespace se puede tener solamente una clase alumno.

3. Explique el **ciclo de vida** de un objeto. Detalle las funciones del **operador new** , del constructor y del **Garbage Collector** .

Se compone de 3 partes:

* Primero se crea el objeto utilizando el operador new (el cual asigna memoria) y se usa un constructor para inicializar el objeto en esa memoria. (new no participa de ninguna manera en la inicialización)
* Segundo se utiliza, se llaman a métodos y atributos
* Tercero se Destruye, la referencia en memoria se pierde (porque termino el programa, se cambió o eliminó la variable, etc.). El Garbage Collector liberará memoria cuando lo crea necesario.

4. ¿En qué **segmento de memoria** se almacenan los tipos valor (value type) y en cuál los tipos por referencia (reference type)? ¿En cuál interfiere al Garbage Collector?

El CLR administra dos segmentos de memoria, Stack y Heap.

Los tipos valor (short, char, double, string, int, etc.) se almacenan en el Stack y los de referencia en el Heap.

El GC administra el segmento de memoria Heap, ya que lo almacenado en el Stack se elimina cuando termina la función que esté utilizando lo almacenado en el mismo.

5. Compare y describa: **Destrucción determinista y no determinista**. Asocie con el concepto de variables y objetos.

* **Determinista:** Una variable local se crea en el momento de declararla y se destruye al final del ámbito (scope) en el que está declarada. Tiene un punto inicial y final de su vida.
* **No determinista:** A diferencia de una variable local, un objeto aparece cuando se crea pero no se destruye al final del scope en el que se crea, o sea, su creación es determinista pero su destrucción no, por lo que tiene tiempos de vida mas largos, ya que un objeto puede existir mucho más allá de una llamada al método.

**Constantes:**

6. ¿Qué es una **constante**? ¿Se puede declarar constantes estáticas (static const)?

Tanto la constante como la estática copian su valor en una variable a la cual se le asigna, la diferencia es que la estática se puede modificar (se modificará su valor en cada constante que este llamándola) y la constante no. Esto por lo que no se puede crear una variable static const.

7. ¿Cuándo se asigna el valor a las constantes (tiempo de compilación o ejecución)? ¿Dos objetos del mismo tipo pueden tener distintos valores en una misma constante? Relacione con atributos de instancia

y estáticos.

El valor es asignado en tiempo de compilación para permanecer inalterable durante el tiempo de ejecución, es por esto que no se pueden obtener en variables objeto, distintos valores de una constante.

**Constructores:**

8. ¿Qué es un **constructor**? ¿Cuál es su función?

Son métodos especiales que se utilizan para inicializar objetos junto con la palabra new.

9. ¿Qué es el **constructor por defecto**? ¿Qué sucede con el mismo cuando declaramos un constructor nuevo en la clase?

Es un constructor público que no retorna nada (ni void), no recibe argumentos e inicializa los campos en cero, false o null.

Si declaramos un nuevo constructor, el anterior (el defecto) se reemplaza por este.

10. ¿Con qué valores se cargan los atributos cuando se llama al constructor por defecto?

Respondido en pregunta 9.

11. ¿Qué es y para qué sirve un **constructor estático**? ¿En qué se diferencia su sintaxis de los métodos de instancia?

Un constructor estático inicializa clases estáticas, y es llamado una única vez, es por esto que no puede acceder a atributos de instancia, no llevan modificadores de acceso y tampoco pueden recibir parámetros.

12. ¿Cuántas veces se puede llamar a un constructor estático? ¿Quién lo puede llamar?

Como se comentó en la respuesta 11, es llamado una única vez por tipo en una sola ejecución del programa. El usuario no puede llamarlo de forma explícita, y se lo llama implícitamente antes de que se cree una instanciación para inicializar la clase.

13. ¿Se ejecutará primero un constructor estático o uno de instancia?

Primero el estático y luego el de instancia.