**HERENCIA**

La herencia (a la que habitualmente se denomina **subclase**) proviene del hecho de que la subclase (la nueva clase creada) contiene las atributos y métodos de la clase primaria. La principal ventaja de la herencia es la capacidad para definir atributos y métodos nuevos para la subclase, que luego se aplican a los atributos y métodos heredados.

Esta particularidad permite crear una estructura jerárquica de clases cada vez más especializada. La gran ventaja es que uno ya no debe comenzar desde cero cuando desea especializar una clase existente. Como resultado, se pueden adquirir bibliotecas de clases que ofrecen una base que puede especializarse a voluntad (la compañía que vende estas clases tiende a proteger las datos miembro usando la [encapsulación](https://es.ccm.net/contents/410-poo-encapsulacion-de-datos)).

**Polimorfismo**

Es la capacidad que tienen ciertos lenguajes para hacer que, al enviar el mismo mensaje (o, en otras palabras, invocar al mismo método) desde distintos objetos, cada uno de esos objetos pueda responder a ese mensaje (o a esa invocación) de forma distinta.

**Abstracción**

La abstracción la podemos definir como el proceso de**identificar aquellas características (atributos) y acciones o comportamientos (métodos) propios de un elemento que deseemos representar.**

La abstracción consiste en aislar un elemento de su contexto o del resto de los elementos que lo acompañan. ... La abstracción encarada desde el punto de vista de la programación orientada a objetos expresa las características esenciales de un objeto, las cuales distinguen al objeto de los demás.

Lo que hace preguntarse, ¿qué es un ADT en programación?

Un tipo de dato abstracto o ADT (Abstract Data Type) es definido por una especificación abstracta, es decir, permite especificar las propiedades lógicas y funcionales de un tipo de dato.

Así, ¿cómo saber si un objeto es una instancia de una clase?

En un lenguaje en el que cada objeto es creado a partir de una clase, un objeto es llamado una instancia de esa clase. Cada objeto pertenece a un tipo y dos objetos que pertenezcan a la misma clase tendrán el mismo tipo de dato. Crear una instancia de una clase es entonces referido como instanciar la clase.

Igualmente, ¿qué significa que la ley es abstracta?

Abstracta e impersonal: las leyes no se emiten para regular o resolver casos individuales, ni para personas o grupos determinados, su impersonalidad y abstracción las conducen a la generalidad. Ignorancia no es excusa: nadie puede invocar su desconocimiento o ignorancia para dejar de cumplirla.

**Encapsulamiento**

Es el proceso de almacenar en una misma sección los elementos de una abstracción que constituyen su estructura y su comportamiento; sirve para separar el interfaz contractual de una abstracción y su implantación.

Consiste en agrupar una líneas de código fuertemente relacionadas, darles un nombre y encapsularlas en un método o una clase.

El encapsulamiento es el divide y vencerás de la programación.

Al separar tu código en pequeños fragmentos que puedas tratar individualmente, liberarás memoria en tu cabeza.

Imagínatelo como un puzle de 400 piezas: mientras lo montas, estás lidiando con muchas variables, pero una vez montado, lo puedes tratar como un todo. Cuando lo miras una vez montado ya no ves las 400 piezas, sino una única imagen: has creado una abstracción. :)

En el momento en que consigas agrupar varios conceptos en uno (a esto se le llama crear una abstracción) habrás liberado tu cerebro de memoria porque ya no necesitarás recordar el resultado final.

**Clases**

Podemos definir informalmente una clase como una plantilla (o esqueleto o plano) a partir de la cual se crean los objetos.

Una clase es una herramienta que tenemos para modelar objetos de programación, lograr que se comporten como queremos y hacer tantas copias de estos objetos como necesitemos.

Por ahora simplemente pensemos que es código que está encapsulado en un Script en programación

Cuando programamos en C# utilizando Unity, al crear un nuevo Script automáticamente creamos una clase cuyo nombre coincide con el nombre del archivo del Script.

La terminología que usamos al referirnos a una clase probablemente cambia dependiendo de la bibliografía o los profesores, pero si logramos captar la idea básica podremos usar este conocimiento para crear soluciones para nuestros proyectos.

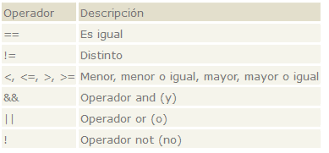
**Atributos**

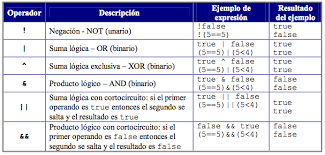
Los atributos son las caracterísiticas individuales que diferencian un objeto de otro y determinan su apariencia, estado u otras cualidades. Los atributos se guardan en variables denominadas de instancia, y cada objeto particular puede tener valores distintos para estas variables. **Objetos**

 Representa alguna entidad de la vida real, es decir, alguno de los objetos únicos que pertenecen al problema con el que nos estamos enfrentando, y con el que podemos interactuar.

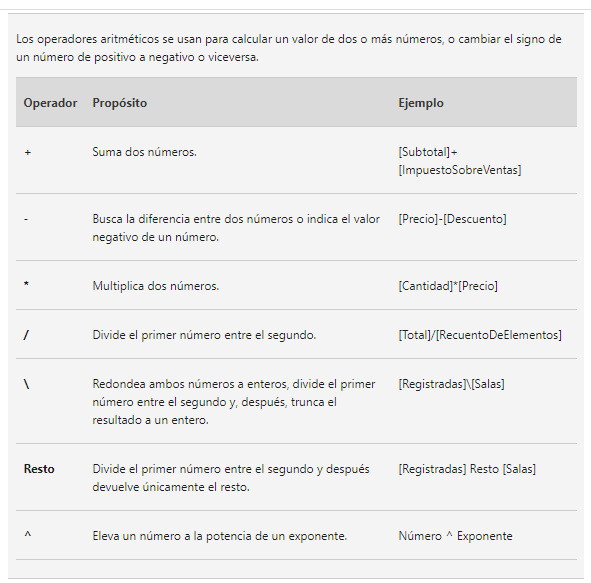
Cada objeto, de igual modo que la entidad de la vida real a la que representa, tiene un estado (es decir, unos atributos con unos valores concretos) y un comportamiento (es decir, tiene funcionalidades o sabe hacer unas acciones concretas).

**Operadores lógicos**





OPERADORES ARITMETICOS



**¿Qué es if else anidado?**

if anidado: un if anidado es una sentencia if que esta dentro de otro if o else. Sí, JavaScript nos permite anidar las sentencias if dentro de las sentencias if, es decir, podemos colocar una sentencia if dentro de otra sentencia if.

¿Qué es un IF anidado en C++?

Estructura de Control if (anidada) Las sentencias if implementan decisiones que implican una o dos alternativas. Una sentencia if es anidada cuando la sentencia de la rama verdadera o la rama falsa es a su vez una sentencia if.

Sintaxis:

          if (condición1)

                  estatuto1

             else

                  if (condición2)

                        estatuto2

                                            •

                                            •

                                            •

                           else

                       if (condiciónn)

                             estatuton

                                  else

                              estatutoe

Si tenemos el siguiente fragmento de código:

      if (condición 1)  
      if (condición 2)  
              estatuto;  
      else               // este else pertenece al if de la condición 2, pues se

                                // se asocia al if más cercano         
           estatuto;

Si queremos que el else pertenezca al primer if debemos poner:

      if (condición 1)  
      {  
         if (condición 2)  
               estatuto;  
      }  
      else               // con el uso de llaves cerramos el if anidado y el else  
         estatuto;     // pertenece al primer if

Que es migraciones?  
Php artisan migration

Las migraciones **son como el control de versión para tu base de datos, permiten que tu equipo modifique y comparta el esquema de base de datos de la aplicación**. Las migraciones son emparejadas típicamente con el constructor de esquema de Laravel para construir el esquema de base de datos de tu aplicación

Si inclusive has tenido que decirle a un miembro de equipo que agregue una columna manualmente a sus esquemas de bases de datos local, has encarado el problema que solucionan las migraciones de base de datos.

La clase [facade](https://styde.net/laravel-6-doc-facades/) Schema de Laravel proporciona soporte de base de datos orientado a la programación orientada a objetos para la creación y manipulación de tablas a través de todos los sistemas de bases de datos soportados por Laravel.

Generando migraciones

Para crear una migración, usa el [comando Artisan](https://styde.net/laravel-6-doc-consola-artisan/) make:migration:

php artisan make:migration create\_users\_table

La nueva migración estará ubicada en tu directorio database/migrations. Cada nombre de archivo de migración contiene una marca de tiempo, la cual permite que Laravel determine el orden de las migraciones.

Las opciones --table y --create también pueden ser usadas para indicar el nombre de la tabla y si la migración estará o no creando una nueva tabla. Estas opciones rellenan previamente el archivo stub de migración generado con la tabla especificada:

php artisan make:migration create\_users\_table --create=users

php artisan make:migration add\_votes\_to\_users\_table --table=users

Si prefieres especificar una ruta de directorio de salida personalizada para la migración generada, puedes usar la opción --path al momento de ejecutar el comando make:migration. La ruta de directorio dada debe ser relativa a la ruta de directorio base de tu aplicación.

Puede usar el make:migration [comando Artisan](https://laravel.com/docs/9.x/artisan) para generar una migración de base de datos. La nueva migración se colocará en su database/migrationsdirectorio. Cada nombre de archivo de migración contiene una marca de tiempo que permite a Laravel determinar el orden de las migraciones:

php artisan make:migration create\_flights\_table

Laravel usará el nombre de la migración para intentar adivinar el nombre de la tabla y si la migración creará o no una nueva tabla. Si Laravel puede determinar el nombre de la tabla a partir del nombre de la migración, Laravel completará previamente el archivo de migración generado con la tabla especificada. De lo contrario, simplemente puede especificar la tabla en el archivo de migración manualmente.

Si desea especificar una ruta personalizada para la migración generada, puede usar la --pathopción al ejecutar el make:migrationcomando. La ruta dada debe ser relativa a la ruta base de su aplicación.