1.-Características del paradigma orientado a objetos.

Es un paradigma de programación que usa objetos y sus interacciones, para diseñar programas informáticos usando técnicas, como la herencia, abstracción, polimorfismo y encapsulamiento. Surge en la historia como un intento para dominarla complejidad que, de forma innata, posee el software

2

* **C++**
* **PHP (a partir de su versión 5)**
* **Visual FoxPro (en su versión 6)**
* [**XBase++**](http://es.wikipedia.org/wiki/XBase%2B%2B)
* **Java**

3

En 1980 surge C++ de la mano de Bjarne Stroustrup (también de Laboratorios Bell de AT&T) con el objetivo de añadir a C nuevas características: clases y funciones virtuales (de SIMULA67), tipos genéricos y expresiones, la posibilidad de declarar variables en cualquier punto del programa, y sobre todo, un auténtico motor de objetos con herencia múltiple que permite combinar la programación imperativa de C con la programación orientada a objetos.

El siguiente hecho fundamental en la evolución de C++ es sin duda la incorporación de la librería STL años más tarde, obra de Alexander Stepanov y Adrew Koening. Esta librería de clases con contenedores y algoritmos genéricos proporciona a C++ una potencia única entre los lenguajes de alto nivel.

4.-

* Bjarne Stroustrup desarrolló el lenguaje C++
* Anders Hejlsberg diseño el lenguaje C#
* Dennis Ritchie diseñó el lenguaje C
* Ken Thompson desarrolló el lenguaje B

5.-

Las características que debe tener un lenguaje de programación para ser orientado a objetos son:

* **Abstracción:** Denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos y cuando lo están, una variedad de técnicas son requeridas para ampliar una abstracción. El proceso de abstracción permite seleccionar las características relevantes dentro de un conjunto e identificar comportamientos comunes para definir nuevos tipos de entidades en el mundo real
* **Encapsulamiento:** Significa reunir a todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión de los componentes del sistema.
* **Modularidad:** Se denomina Modularidad a la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes. Estos módulos se pueden compilar por separado, pero tienen conexiones con otros módulos.
* **Principio de ocultación:** Cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una interfaz a otros objetos que especifica cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas, solamente los propios métodos internos del objeto pueden acceder a su estado. Esto asegura que otros objetos no pueden cambiar el estado interno de un objeto de maneras inesperadas, eliminando efectos secundarios e interacciones inesperadas
* **Polimorfismo:** Comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre, al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica.
* **Herencia:** Las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo.
* **Recolección de basura:** Es la técnica por la cual el entorno de objetos se encarga de destruir automáticamente, y por tanto desvincular la memoria asociada, los objetos que hayan quedado sin ninguna referencia a ellos. Esto significa que el programador no debe preocuparse por la asignación o liberación de memoria, ya que el entorno la asignará al crear un nuevo objeto y la liberará cuando nadie lo esté usando.