**INSTITUTO TECNOLOGICO DEL SUR DE NAYARIT**

**Ingeniería en Tecnologías de la Información y Comunicaciones**

**Programación Orientada a Objetos**

**Interfaces**

**Docente: Ing. Cinthia Anahí Mata Bravo**

**Alumno: Josval de Jesús Ceceña Meza**

**No. Control: 191140044**

**2do Semestre**

**19/03/20**

**Índice**

**Interfaces c# --------------------------------------- 3**

**Resumen -------------------------------------------- 4**

* **Anotaciones--------------------------------- 4**

**Interfaces en C#**

Es un medio común para que los objetos no relacionados se comuniquen entre si. Por ejemplo en Java (donde se le llama Interfaz (Java interfaces)), la interfaz comparable especifica un método comparar con () que las clases deben implementar. Esto significa que un método aparte de ordenamiento, por ejemplo, puede ordenar cualquier objeto que implemente la interfaz comparable, sin tener que saber nada sobre la naturaleza interna de la clase. Pueden tener métodos y atributos pero no se pueden poner funcionalidad. En programación orientada a objetos decimos que la interfaz de una clase define que podemos y que no podemos hacer con objetos de dicha clase.

Se conoce como la interfaz de usuario al **medio que permite a una persona comunicarse con una máquina**. La interfaz, en este caso, está compuesta por los puntos de contacto entre un usuario y el ordenador. Además del ejemplo de ratón mencionado, otra interfaz de este tipo es la pantalla del [monitor](https://conceptodefinicion.de/monitor/) o el teclado.

Una interfaz en [Java](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)) es una colección de métodos abstractos y propiedades constantes.

En las interfaces se especifica qué se debe hacer pero no su implementación. Serán las clases que implementen estas interfaces las que describen la lógica del comportamiento de los métodos.

La principal diferencia entre interface y abstract es que un interface proporciona un mecanismo de encapsulación de los protocolos de los métodos sin forzar al usuario a utilizar la herencia.

**RESUMEN**

Una interface en c# se refiere a un constructor de lenguaje que nos permite llevar cosas y hacer algo con ellas, la interface no puede llevar métodos concretos, ni declaración de variables, nos sirve para decir una serie de comportamientos que puede tener una clase. La interface nos sirve para crear conceptos.

Una interface es parecida a una clase abstracta, mas no son iguales, la clase abstracta se utiliza para definir subclases que heredan los métodos abstractos que nosotros declaramos. Una interfaz, por otro lado, todos los métodos son abstractos, las interfaces se implementan y no se heredan, esa sería la primer diferencia de la clase abstracta.

En base a los videos e información que investigue, hice varias anotaciones para un mayor entendimiento del concepto.

**Anotaciones**

* Una interface en programación orientada a objetos
* Con un interface si se puede implementar dos interfaces
* Una interface es parecido a una clase abstracta
* Es un contrato el cual las clases deben cumplir con esta
* Se tiene que crear a partir de una clase que ya implemento los métodos
* Las interfaces pueden tener métodos y atributos pero no se pueden poner funcionalidad
* No se puede crear un objeto directamente de un interface
* Se tiene que crear un objeto a partir de los métodos de la interfaz
* Toda clase tiene una interfaz.
* Las interfaces se utilizan más para definir el comportamiento de los objetos en la jerarquía
* Una interface no es una clase