1/20/23 - **Herencia**

|  |  |
| --- | --- |
| Puntos clave | Notas |
| Cuando detecto características y comportamientos iguales, entonces significa que debo realizar una abstracción.  **Don’t repeat yourself** es una filosofía que promueve la reducción de duplicación en programación, esto nos va a inculcar que no tengamos líneas de código duplicadas.  Toda pieza de información nunca debería ser duplicada debido a que incrementa la dificultad en los cambios y evolución | **Don’t repeat yourself** es una filosofía que promueve que no tengamos líneas de código duplicadas  Debe considerarse al momento de llevar a cabo el análisis, de lo contrario estarias infringiendo el principio antes mencionado.  Por lo tanto, toda pieza de información nunca debería ser duplicada ya que:   * La **duplicación** incrementa la dificultad en los cambios y su evolución. * De mantenerlo, **va a dificultar** que en el futuro nosotros queramos **ejecutar cambios o incluso involucrar un objeto más**. * Hará que el código sea un poco **más difícil de leer**. * Un poco **más difícil de entender** y por supuesto tener **mantenimiento** sobre él se nos va a complicar bastante.   La **reutilizacion**: es uno de los principios de la programación orientada a objetos.  La **herencia**: es una pieza clave para reutilizar líneas de código al más no poder.   * La herencia nos permitirá crear nuevas clases a partir de otra. * una vez que nosotros detectemos elementos duplicados empezaremos a ejecutar un análisis. * Llevar a cabo una abstracción, de tal manera que puedas generar una clase que sea la más general nos permita crear nuevas clases. * Por eso es que la herencia nos permitira crear nuevas clases a partir de otras. * Permite simular modelos de la vida real, va a tratar de simular escenarios en en la vida real para poderlos pasar al código. * **Por ejemplo**: como cuando tú puedes recibir una herencia cuando tu padre te puede heredar un auto te puede heredar cualquier otra cosa exactamente igual así va a funcionar la herencia |
| Sumario: La **herencia** nos permite crear nuevas clases a partir de otras, se basa en modelos y conceptos de la vida real. También tenemos una jerarquía de padre e hijo. | |

1/20/23

|  |  |
| --- | --- |
| Puntos clave | Notas |
|  | En la **herencia** también tendremos una jerarquía y tendremos una una estructura de padre e hijo.  Un padre puede tener muchos hijos no solamente uno es muy común que un padre tenga un solo hijo pero en general y justamente en nuestro proyecto veremos que tendremos bastantes hijos a partir de la abstracción que haremos.  **A picture containing graphical user interface  Description automatically generated**  **La clase padre en rojo** es esa clase en programación orientada a objetos como la superclase usada para referirse siempre a la clase primigenia o principal de la cual parte o derivan otras llamadas subclases.  Las subclases entonces, de acuerdo esta jerarquía, son aquellas que derivan de la superclase, siendo estas consideradas sus hijas.  Un ejemplo como podríamos ejercitarnos para identificar herencia es precisamente es en este caso tenemos 3 clases.    **Futbolista, entrenador y masajista** si nosotros analizamos estas 3 **clases** observamos que ellas tienen en común cuatro atributos:  **el identificador > nombre > apellidos > edad**  además también tienen en común algunos **métodos** como: **concentrarse > viajar** |
| Sumario: | |

1/4/23

|  |  |
| --- | --- |
| Puntos clave | Notas |
| Cuando estas heredan, significa que automáticamente se tienen:   * Los métodos que están contenidos en la superclase serán heredados a las clases hijas sin necesidad de escribirlo en el código, simplemente aplicando la herencia. | A picture containing diagram  Description automatically generated   * En programación orientada a objetos, si detectamos que hay elementos repetidos esto me indica que tengo que hacer algo. * Que algo no está bien y que seguramente si permanece de esta manera, seguramente me va a traer problema. * Cuando se detecta una relación de estos elementos, se puede generaruna abstraccion de eso y entonces crear una clase que tenga todos estos elementos en común.   Diagram  Description automatically generated |
| Sumario: | |