|  |  |  |
| --- | --- | --- |
|  | **Parcial #1**  Desarrollo de Software Web | Código: |
| Versión 01 |
| Edición: 17/03/2021 |

La empresa botanical S.A requiere llevar un control de las diferentes especies de plantas que maneja con el fin de tener información disponible de las mismas entre las plantas que tienen manejan lechugas, moras y zanahorias con los siguientes datos

* Nombre
* color
* edad (lechugas 1 mes, zanahoria 3 meses y moras 6 meses)

1. Muestre el listado de plantas (con al menos 2 lechugas)en formato xml (se deben respetar los conceptos de programación orientada a objetos; Encapsulación, polimorfismo, herencia, abstracción), realizar una distribución de paquetes teniendo en cuenta el patrón MVC. (3)

* Funcionalidad (.5)
* POO (1)
* Codificación y Patrón usado (.5)
* Realice un diagrama de clases y uno de paquetes para la aplicación descrita (.5)
* Suba el código al git en la rama master (0.25)
* Suba el diagrama en la rama dev (0,25)

1. Trasformar el siguiente xml en Json (.5)
2. Explique los principios solid(.5)
3. Dada la clase

Public class Bicicleta {

Public Bicicleta(){

}

Public void armarbicicleta(){

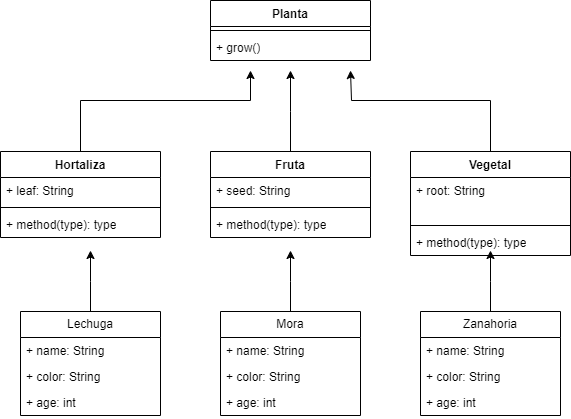
unir(armarRuedas(), armarChasis());

}

Public String armarChasis(){}

Public String armarRuedas(){}

}

Refactorizar para que cumpla principio single responsability

2)

JSON:

{“Mora”: { “name”: “americana”, “color”: “morado”, “AGE”: 6}}

3) solid tiene 5 principios:

1. Responsabilidad unica: que las clases y métodos se encargen de solo una cosa

2. Open-close: abierto a crecer pero cerrado a cambios

3. sustitución: las clases hijas puden ser las clases padres es decir tener sus atributos

4. segregación de interfaz : se deben utilizar interfases o (muchas) en vez de clases y estas interfases deben cumplir con cosas espeficias

5. de dependencia: preferir interfases y clases abtactas que claes normales

4)

Public class Bicicleta {

Public Bicicleta(){

}

Public void armarChasis(){}

Public armarChasis(){}

}