|  |  |
| --- | --- |
| **Integrantes** | |
| <nombres y apellidos> | <código> |
| <nombres y apellidos> | <código> |
| **Repositorio Bitbucket** | |
|  | |

# Estructura máxima 7 HOJAS

# Introducción

Realice una breve explicación de que hicieron y cómo llegaron a la solución propuesta.

# Requerimientos Funcionales

Identifique los 3 requerimientos funcionales principales para el sistema propuesto y documéntelos siguiendo el siguiente formato:

|  |  |
| --- | --- |
| Nombre |  |
| Resumen |  |
| Entradas | |
|  | |
| Resultados | |
|  | |

# Modelo del mundo

Presente de forma clara su modelo del mundo usando la notación UML y explique los diferentes elementos del que consiste (para cada parte del proyecto). **Indique que cambios se realizaron desde la primera entrega y el porqué de ellos**.

# Diseño, implementación y pruebas de las estructuras de datos

Indique como **las estructuras que usted creó,** las cuales deben expresarse en el modelo UML, satisfacen los requerimientos funcionales propuestos por el cliente al igual como los no funcionales. **Indique que cambios se realizaron desde la primera entrega y el porqué de ellos**.

Recuerde que en Bitbucket debe quedar la implementación de las estructuras definidas anteriormente. Para ello, cree una rama con la siguiente estructura de nombramiento:

***entrega\_final\_dd\_mm\_yy\_HH\_MM\_SS (reemplazan por su fecha y hora de subida)***

Es importante que su código cumpla con las siguientes restricciones:

* Cada estructura debe contar con pruebas unitarias
* Las estructuras deben ser genéricas
* Debe ser coherente el diseño con la implementación realizada
* **No** puede **usar ni extender** las **estructuras** que provee **por default Java**. Si usted usa alguna implementación de estructuras, que se encuentren en la web o literatura, debe indicar la procedencia y dar los respectivos créditos, tanto en la documentación del código como en este documento.

# Importación de la información

Usted debe crear, basado en los archivos JSON entregados con anterioridad, un set de datos que represente el comportamiento del sistema en **una semana**. Es decir, debe crear datos considerables de dueños y mascotas, al igual que trazas para los eventos diarios que se deben generar. Dichos archivos deben quedar también en el Bitbucket (carpeta data), como se explicó en el punto anterior.

Por otra parte, se debe indicar como fue el procedimiento/rutina para la importación de los archivos JSON (en este documento) y dejar la codificación en el Bitbucket. El código implementado para la carga de información también de contar pruebas unitarias.

**Indique que cambios se realizaron desde la primera entrega y el porqué de ellos**.

# Implementación del sistema

Debe quedar la implementación completa del sistema tal como se indicó en el enunciado del proyecto (parte A, B y C). Recuerde que su código debe quedar en Bitbucket en una nueva rama con la siguiente estructura de nombramiento:

***entrega\_final\_dd\_mm\_yy\_HH\_MM\_SS (reemplazan por su fecha y hora de subida)***

En este documento no olvide **indicar quién hizo qué**.

# Análisis de complejidad

De los requerimientos propuestos tome los 3 más importantes, copie el fragmento de código que satisface dicho requerimiento y haga un análisis de complejidad para cada uno. Indique el porqué de su respuesta, si piensa que no es la solución óptima explique porqué tomó dicha decisión y cual podría ser una mejor solución (si es la óptima indique porqué lo es).