# DASDA ASSIGNMENT

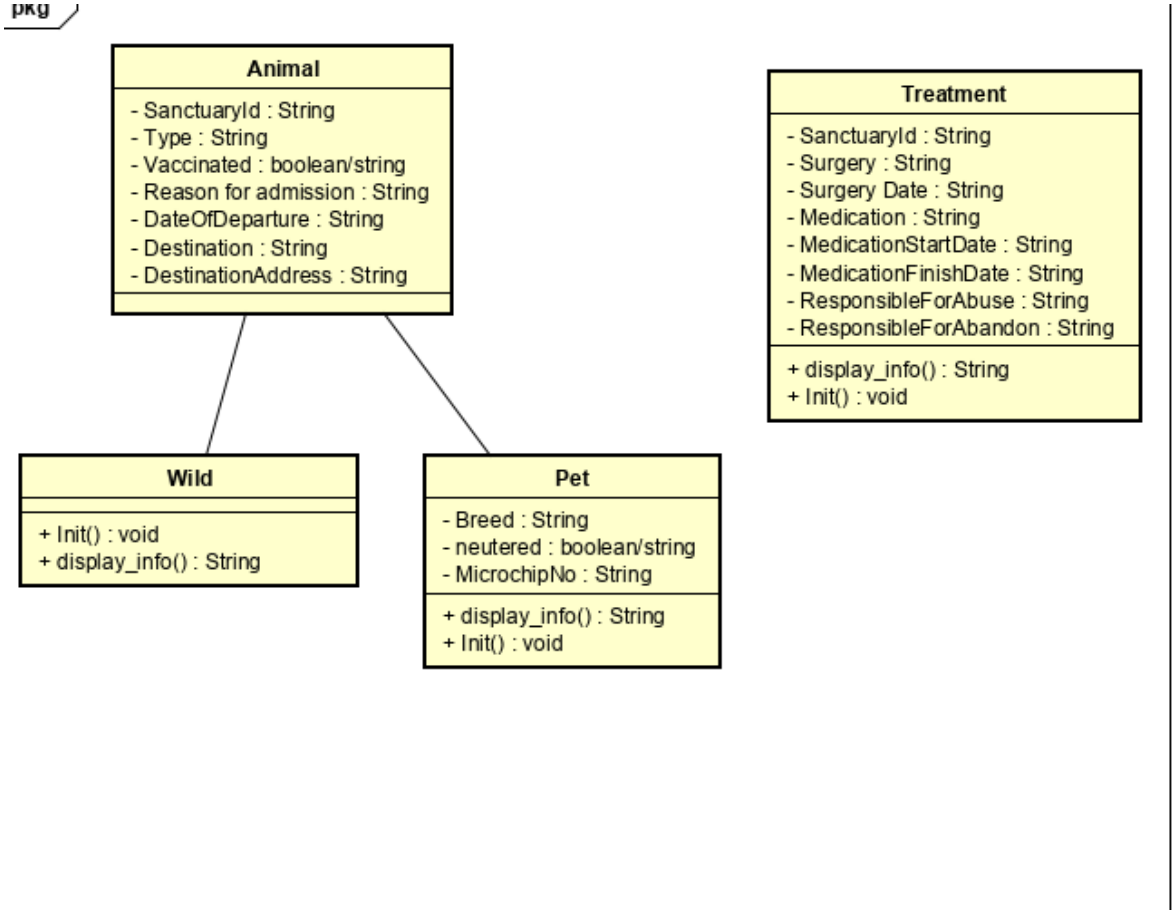
## Introduction

Looking at the data set sets we have been provided, I can see that each animal contains contains a unique ID as well as a set of information that goes with each specific animal. We would never be able to have two animals with the same ID or it would not work. My thought process upon seeing this was to use a class object system and an object list. I read in the csv as a dictionary in order to make a clear layout of keys + values so then when it came to creating the objects, the attributes would be initialised based on the key and value pair that the dictionary held and each object would be put into the appropriate object list. Where each row in the list would be the set of attributes for an object. That way when we need to search for any information about a specific animal, we can look for the index of the unique sanctuary id and then print out all of the attributes that are paired to the ID of that object (For example pet_list[index].sancatuaryid). Storing this object data in a list means we can easily loop through and navigate when we are required to search or change values.

The first idea I had was to simply put all of the data into a regular list. I had tested regular list for the data without a class object model, but upon adding the data to a csv file from the list, it became hard to dissect the information that I wanted to as calling for an index of the list e.g. list[1] ending up printing the entire row of values instead of just the 1 key value that I wanted since each piece of data in the list was an entire row. I decided to this solution wasn't appropriate for me to use as it seemed that wanting to edit a singular data value down the line would prove rather troublesome.

Based on the class object model I decided to use, creating new animals would be a rather simple task. All we would need to do is create a new instance of the class, assigning values to each attribute it is supposed to contain and creating a new object. We could then easily add this to our csv file, since the sanctuary potentially will be housing lots of animals, keeping this process simple is of a high priority. For producing the required lists of information to present to the user I decided to use lists. Creating a list and then adding the Id attribute for each animal object that met the necessary conditions. Since the lists needed to come in a specific order, whether alphabetical or ascending number, using a list and then an appropriate algorithm to manipulate the data was what I decided to do. I used a bubble sort algorithm in order to swap the values until the required order was met, while possibly not the quickest algorithm, it doesn't lead to much room for error and is very accurate.

## Initial Class Diagram Example




## Pseudocode

### Reading CSV Files

```
READ file
    for (till end of csv) {
        take in values for (Id, type, breed, vaccinated, neutered, microchip,
admission,
        arrival_date, departure_date, destination, destination address)
        Assign them to list as instance of Animal class
    }
```

### Writing to CSV Files

```
OPEN file
WRITE header values - Key Column (Id etc)
For pet/wild/treatment in pet/wild/treatment list
    WRITE row (each object attribute to the correct column)
```

### Menu Navigation

```
WHILE true
    PRINT Welcome to animal sanctuaryid
    PRINT list of options (add data, view info, edit data, exit)
    INPUT menuchoice
    IF add data
        CALL new arrival function
    ELIF view info
        CALL list view function
    ELIF edit data
        CALL edit list function
    ELIF exit
        exit
```

**New Arrival Function**

```
IF new animal
    INPUT pet or wild
    IF PET
        CALL new pet function
    ELIF wild
        CALL new wild function
    ELSE
        PRINT Invalid
        recall function
IF new surgery
    CALL new surgery function
```

**New Pet Function**

```
OPEN file to append
    INPUT values for new pet
    VALIDATE make sure new ID is unique and has correct format
    VALIDATE make sure new chip number has correct format
    WRITE new row in csv
```

**New Wild Function**

```
OPEN file to append
    INPUT values for new wild animal
    VALIDATE make sure ID is unique and has correct format
    WRITE new row in csv
```

**New Surgery Function**

```
OPEN file to append
    INPUT values for new surgery
    VALIDATE make sure ID is unique and correct format
    WRITE new row in csv
```

**View List Function**

```
PRINT list of available data to view (Animal data, dogs, cats, owner, abuse,
abandon)
INPUT user choice of list to view
```

```
IF animal data
    CALL animal data function
ELIF dogs
    CALL dog list function
ELIF cats
    CALL cat list function
ELIF owner
    CALL owner function
ELIF abuse
    CALL abuse function
ELIF abandon
    CALL abandon function
ELSE
    PRINT invalid, return to menu
    CALL menu
```

**Animal_Data Function**

```
INPUT pet or wild
IF wild
    INPUT ID
    FOR all values in wildlist
    IF id found
        PRINT CALL display info function
IF pet
    INPUT ID
    FOR all values in petlist
    IF id found
        PRINT CALL display info function
ELSE
    PRINT invalid return to menu
    CALL menu function
```

**Dog_List and Cat_List Functions**

```
PRINT Cats/Dogs ready for adoption
    FOR everything in list of pets
        IF CAT/DOG and VACCINATED and NEUTERED and CHIPPED
    PRINT CAT/DOG ID
```

**Return to Owner**

```
CREATE temp list
FOR all values in pet list
    IF reason for admisson EQUAL TO Lost OR Car Accident
    ADD to temp_list
FOR all values in wild list
    IF reason for admission EQUAL TO Car Accident
    ADD to temp_list
PRINT List of animals ready to be returned
CALL Bubble Sort function in order to sort in ascending order
```

**Abuse and Abandon functions**

```
CREATE templist, abuse/abandon list
FOR all values in pet list
    IF abuse/abandon NOT empty
        ADD to temp list
FOR value in templist
    IF value NOT in abuse/abandon list
        ADD to abuse/abandon list
PRINT List of people who have abused/abandoned animals
CALL Bubble sort function to put them in alphabetical order
```

**Bubble Sort**

```
swap EQUAL true
WHILE swap
    swap EQUAL False
    FOR all values in list
        IF value1 > value2
        change positions
        swap = true
PRINT list
```

**Edit List**

Based on the data that we have been given I would consider the status of the animal to be whether or not it is still present in the sanctuary at this current time. Meaning that by editing the date of departure, the destination of the departure as well as the address of the destination, you are editing the status of the animal.

```
PRINT list of things to edit(Chip No. Neuter status, date of departure,
destination, Surgery)
IF chip
    CALL chip edit function
IF neuter
    CALL neuter edit function
IF departure date
    CALL departure date edit funciton
IF destination
    CALL destination edit function
IF surgery
    CALL surgery edit function
```

**Chip/Neuter Edit Functions**

```
INPUT Id of animal you want to change value for
FOR all values in pet list
    IF pet ID is found
        INPUT Chip number/Neuter status
        change to new value
        PRINT new info of animal
        write to csv
```

**Departure Date Function**

```
INPUT ID of animal
```

```
IF first value of ID is P
    FOR values in pet list
        IF match
            INPUT date of departure
            change to new value
            print new info of pet
            write to csv
IF first value of ID is W
    For values in wild list
        IF match
            INPUT date of departure
            change to new value
            print new info of wild
            write to csv
```

**Destination function**

```
INPUT ID of animal
IF first value of ID is P
    FOR values in pet list
        IF match
            INPUT destination
            INPUT destination address
            change to new values
            print new info of pet
            write to csv
IF first value of ID is W
    For values in wild list
        IF match
            INPUT destination
            INPUT destination address
            change to new values
            PRINT new info of wild
            write to csv
```

**Surgery_Edit function**

```
INPUT id of animal
FOR all values in treatment_list
    IF match found
        INPUT values(surgery type, date of surgery, medication, medication start
and finish dates)
        IF any input values are left blank do not change
        ELSE update to new values
        PRINT new info
        write to csv
```

Main Class

```
class Animal
    Declare class attributes
        id
        type
        vaccinate
        admission
        arrive date
```

```
        depart date
        destination
        address

        INITIALISE values
class Treatment
    Declare class attributes
        id
        surgery
        surgery date
        medication
        medication start date
        medication finish date
        abuse
        abandon

        INITIALISE values

        FUNCTION
```

SubClasses - Pet and Wild

```
class Pet (subclass of Animal)
    Declare uniques attributes
        breed
        neutered
        micro

        INITIALISE values

        FUNCTION display info
            string = all class attributes

            return string
class Wild (subclass of Animal)
        INITIALSE values

        FUNCTION display info
            string = all class attributes

            return string
```