

**Developer**: Jesse Jayne

**Date**: 09/21/2021

# IT 145 Global Rain Summary Report Jesse Jayne

## Pseudocode

IMPLEMENT PET CLASS & CHECK-IN METHOD

**Begin** –

**DEFINE Public CLASS Pet** –

INITIALIZE Private Variables

Pet Type;

Pet Name;

Pet Age;

Dog Spaces;

Cat Spaces;

Days Stay;

Amount Due;

**Default CONSTRUCTOR for CLASS Pet** –

Pass parameters to INITIALIZE Public Variables for use in methods CHECK-IN and CHECK-OUT

THIS (Pet)

Pet Type;

Pet Name;

Pet Age;

Dog Spaces;

Cat Spaces;

Days Stay;

Amount Due;

**DEFINE Public METHOD Check-In** –

SET initial values for SPACES variables

GET User Input for Pet Type

IF Pet Type == CAT

THEN check cat space, assign, and GET userinfo for CAT

ELSE IF Pet Type == DOG

THEN check dog space, assign, GET userinfo for DOG

ELSE invalid Pet Type, RETURN Error message and exit

OUTPUT Response and GET additional userinfo

**METHOD ACCESSORS AND MUTATORS** –

DEFINE Public METHODs GET AND SET FOR {

CheckOut

GET Pet CREATE Pet

UPDATE Pet

GET Pet Type SET Pet Type

GET Pet Name SET Pet Name

GET Pet Age SET Pet Age

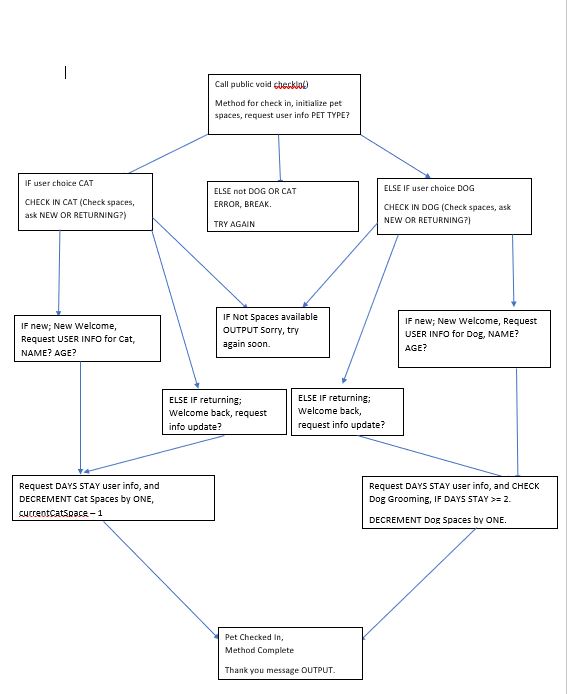
GET Pet Spaces SET Pet Spaces

GET Days Stay SET Days Stay

GET Amount Due SET Amount Due

**END**

## Flowchart – Check-in Method Process

****

## OOP Principles Explanation

Throughout this process object-oriented principals were applied in several ways. First, through the creation of the Pet class itself. This class allows us to utilize the same code for any Dog or Cat a customer chooses to board with us, through another instance of object-oriented design which are the methods. The different methods in the class allow us to collect all of the users information about their pet, and call it back at any time with the same commands for any different pet or pet type. We can also implement inheritance if we choose to with the Check-in and Check-out methods, by passing the check-in parameters to the check-out method, allowing it to use the same variables and data for each unique pet. This type of programming and design will allow us to scale as much as we want within this model, and even expand to additional pet types or services in the future, by just adding additional methods to the already implemented pet class.