

**Developer**: Zachary Nicholas

**Date**: 07/20/2022

# IT 145 Global Rain Summary Report Template

## Directions

Place your pseudocode, flowchart, and explanation in the following sections. Before you submit your report, remove all bracketed text.

## Pseudocode

When you are done implementing the Pet class, refer back to the Pet BAG Specification Document and select either the pet check in or check out method. These methods are detailed in the Functionality section of the specification document.

Write pseudocode that lays out a plan for the method you chose, ensuring that you organize each step in a logical manner. Remember, you will *not* be creating the actual code for the method. You do *not* have to write pseudocode for both methods. Your pseudocode must not exceed one page.

## Variables: Pet Type, pet Name, pet Age, pet weight, pet Spaces (Cat or Dog), Days of stay, amount due.

## Get user inputs for variables

## Ask user to input a pet type in this example dog is the only accepted input, keep prompting user to enter dog or use “Quit’ to exit the program

If dog is entered add one to the current dog spaces and have a new variable “New dog space” hold the value, then check if the new dog space variable has a integer less than or equal to 30, if it is 30 or less continue with the program, if 31 or greater print out “no space available” and end the program.

## Ask for the pets name, pets age, pets weight and, days of stay. Reading them one line at a time to properly assign them to their respective variable names and types

## Once all values have been stored check if the number of days stay is greater than or equal to 2, if yes prompt user to enter ‘Y’ or ‘N’. if yes change the Boolean variable grooming equal to true, if no keep Boolean variable grooming equal to false. Once all of the values are stored then end the program

Save first user input as a string defined as pet\_Name

Prompt user for pet name, pet age, pet weight and, days stay

Continue

Current Dog Space = New Dog Space

Yes

Is new dog space <=30?

New dog space = Current Dog Space+1

Else if “Quit”

If Dog

Prompt user for pet type

Start

Save second user input as a integer defined as pet\_Age

No

Print “No space Available”

Else prompt user to input a valid input

No

Prompt user for Grooming

Yes

Save third user input as a integer defined as pet\_weight

End

If days Stay is >= 2

after all values are entered and stored

Else prompt user to input a valid input

Save fourth user input as a integer defined as days\_stay

No

End

If ‘Y’

Else If ‘N’

Grooming = False

Grooming = True

No

No

## OOP Principles Explanation

Briefly explain how you applied object-oriented programming principles and concepts (such as encapsulation, inheritance, and so on) in your software development work thus far. Your explanation should be one paragraph, or four to six sentences.

With the OOP Principles I have been able to reduce redundant code which will be more apparent in my final project for this when we combine cat and dog. With inheritance I don’t have to make individual variables for both cat and dog type, instead things like name, age, days stay, amount due and spaces available for each pet type, instead we can assign public variables for all of them and let cat and dog use them how they wish. With encapsulation we can make a more straightforward and overall more easy to read code especially with two separate pet types to define and assign their variables. When it comes to abstraction our pet type will be an abstract variable that way we don’t have to define everything that could go into that type of variable. Polymorphism also allows our code to have a more versatile set of a variables in order to make it easier to read and also simplier to code.