

**Developer**: Brennon Fultz

**Date**: 7/21/23

# IT 145 Global Rain Summary Report

## 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.

INPUT pet name

DECLARE DOUBLE Fee

WHILE var less than pet ARRAYLIST length

IF pet name != Pet.name() at ARRAYLIST[var]

INCREMENT var

ELSE IF Pet.daysStay() > 0

IF dog < 20lbs

Fee += 24.00 \* days

IF groomed

Fee += 19.95

RETURN FEE

ELSE IF dog >= 20lbs and < 30lbs

Fee += 29.00 \* days

IF groomed

Fee += 24.95

RETURN FEE

ELSE IF dog >=30lbs

Fee += 34.00 \* days

IF groomed

Fee += 29.95

ELSE

FEE = 18.00 \* days

RETURN Fee

SpaceNumber = 0

ELSE PRINT “Pet not found”

## Flowchart

Based on the pseudocode you wrote, create a flowchart using a tool of your choice for the method you selected. In your flowchart, be sure to include start and end points and appropriate decision branching, and align the flowchart to the check in or check out process. Your flowchart must be confined to one page.



## 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.

The four main principles of object-oriented programming were used to guide this software development work. Abstraction is the idea of keeping unnecessary details hidden from the user, which is why the Pet class is used to generate Pet objects instead of having all the details of the Pet declared explicitly in the main method. Similarly, encapsulation protects an object’s state by keeping certain methods private and this is something which could be implemented in the specific Cat or Dog classes or changed in Pet if there is a need to protect certain attributes. Inheritance will be used when the Dog and Cat classes are created as subclasses of the Pet class, which will allow the code from Pet to be reused. Finally, polymorphism can also be implemented in Cat and Dog, allowing them to use overloaded methods from Pet to perform the same function but with slightly different implementation as needed for their class.