

**Developer**: Mao Christie

**Date**: February 4, 2024

# 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.

Checkout method:

* **Find the pet and its boarding space**
  + **If the pet is a dog,**
    - **if grooming is set to yes/true**
      * **If weight >= 30, add $29.95 + (daysStay \* 34) to amountDue**
      * **If the weight is between 20 and 30 pounds, add $24.95 + (daysStay \* 29) to amountDue**
      * **If weight < 20, add $19.95 + (daysStay \* 24)**
    - **Else get daysStay for the dog**
      * **If weight >= 30, add ($34 \* daysStay) to amountDue**
      * **Else If weight is between 20-30 pounds, add ($29 \* daysStay) to amountDue**
      * **Else add ($24 \* daysStay) to amountDue**
    - **Mark dogSpaceNumber as vacant**
  + **If pet is a cat** 
    - **Add (daysStay \* $18.00) to amountDue**
    - **Mark catSpaceNumber as vacant**

## 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.

Find pet and spaceNumber

If pet is a cat

If pet is a dog

Add daysStay \* $18.00 to amoundDue

If grooming is set to yes/true

Mark catSpaceNumber as vacant

Else, get daysStay for the dog

Else, add (19.95 + (daysStay \* 24)) to amountDue

If weight 20-30, add (24.95 + (daysStay \* 29)) to amountDue

If weight is >= 30, add (29.95 + (daysStay \* 34)) to amountDue

Else add daysStay \* 24 to amountDue

If weight 30-20, add daysStay \* 29 to amountDue

If weight >= 30, add daysStay \* 34 to amountDue

Mark dogSpaceNumber as vacant

## 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.

I used OOP principles and concepts by incorporating private fields in my code for the pet class. By using private fields, I am also applying encapsulation. While writing the pseudocode, I am also referencing and using fields like daysStay which is part of the pet class to calculate the total amount due. This is inheritance, as I’m using fields and methods from the parent class (pet) on the child class(dog/cat).