## Global Rain Logo

**Developer**: Jeffrey M. Perkinson

**Date**: September 2019

# 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.

|  |
| --- |
| PROGRAM checkIn:  function checkIn():  PRINT “What type of pet do you have?”  INPUT the customer’s pet type    WHILE the customer's pet type is not cat or dog  PRINT “Invalid pet type. Please specify cat or dog.”  INPUT the customer’s pet type  ENDWHILE    IF the customer's pet type is a dog THEN  IF available dog space is greater than 0 THEN  RETURN true  ELSE  RETURN false  ENDIF  ELSE  IF available cat space is greater than 0 THEN  RETURN true  ELSE  RETURN false  ENDIF  ENDIF    IF there is space available THEN  PRINT "Has this pet stayed with us before? (Y/N)"  INPUT Y or N    WHILE input is not Y or N  PRINT "Invalid response. Please enter Y or N."  INPUT Y or N  ENDHILE    IF customer is an existing client THEN  UPDATE existing pet information  ELSE  ADD new pet information  ENDIF    PRINT "How many days will the pet be staying with us?"  INPUT the number of days    IF the customer's pet type is a dog THEN  IF the duration of the stay is greater than 2 days THEN  SET grooming to TRUE  ELSE  SET grooming to FALSE  ENDIF  ENDIF    IF the customer's pet type is a dog  SET the dog's space number  DECREMENT available dog space  PRINT dog's space number  ELSE  SET the cat's space number  DECREMENT available cat space  PRINT cat's space number  ENDIF    ELSE  PRINT "No Vacancy."  ENDIF  END. |

## 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/check-out process. Your flowchart must be confined to one page.

|  |
| --- |
|  |

## OOP Principles Explanation

Briefly explain how you applied object-oriented programming principles in the software development process. Your explanation should be one paragraph, or four to six sentences.

|  |
| --- |
| The Pet class (i.e. Pet.java) applies inheritance through its intended purpose to serve as the base class for the derived Cat and Dog classes. Fields and methods shared between the Cat and Dog classes are defined in a single base class to not be duplicated in their definitions. These class attributes are inherited by derived classes that exhibit the same properties and behaviors. This promotes “don’t repeat yourself” (DRY) practices in that shared code is only written once and implemented across varying classes.  Encapsulation is applied through including ‘private’ access modifiers to the class fields, while applying ‘public’ access modifiers to the setter and getter (i.e. write and read) methods. Private access to the member fields prohibits outside modification through simple assignment and reduces the risk of unintended consequences through inadvertent modification. Public access to member methods exposes the ability to read and write to member fields from outside classes but ensures an explicit method call must be made to do so. Encapsulation is one of the fundamental principles of the object oriented programming paradigm, as it promotes the containing of data to a single unit.  Polymorphism is applied through the definition of multiple constructors in the Pet class. There is the default, no argument, constructor and a parameterized constructor defined with several parameters. The constructor is overloaded through varying definitions, the appropriate one of which is bound to the call at compile time based on the arguments provided in the object instantiation call. Abstraction is not applied.  Conditional logic is applied through multiple if-else statements in the checkIn() method. Specific branches in the method are executed based on the input provided. For example, if the type of pet to be checked in is a dog, conditional logic dictates that the available dog space be queried for vacancy availability and available cat space is deemed inconsequential. The reverse occurs if the pet to be checked in is a cat. Conditional logic is also used to determine whether the pet needs groomed while lodging at Pet BAG based on the duration of the stay and whether the pet is a dog. In sum, conditional logic is used to control the flow of the program based on the input provided. |