Table of Contents

[Diagrams 1](#_Toc481342268)

[JSP 1](#_Toc481342269)

[Data Dictionary 1](#_Toc481342270)

[Global 1](#_Toc481342271)

[Local 1](#_Toc481342272)

[Structures and features 2](#_Toc481342273)

[Loops 2](#_Toc481342274)

[Conditional Statements 2](#_Toc481342275)

[Switch and Case 2](#_Toc481342276)

[If/Else 2](#_Toc481342277)

[Finite-State Machine 2](#_Toc481342278)

[Pseudocode 3](#_Toc481342279)

[Load Customer First Name 3](#_Toc481342280)

[Load Customer Surname 3](#_Toc481342281)

[Load Animal Name 3](#_Toc481342282)

[Load Animal Gender 3](#_Toc481342283)

[Load Animal Type 3](#_Toc481342284)

[Load Animal Age 3](#_Toc481342285)

[Load Animal Castration Info 3](#_Toc481342286)

[Load Animal Accident Info 3](#_Toc481342287)

[Calculate Animal Insurance 3](#_Toc481342288)

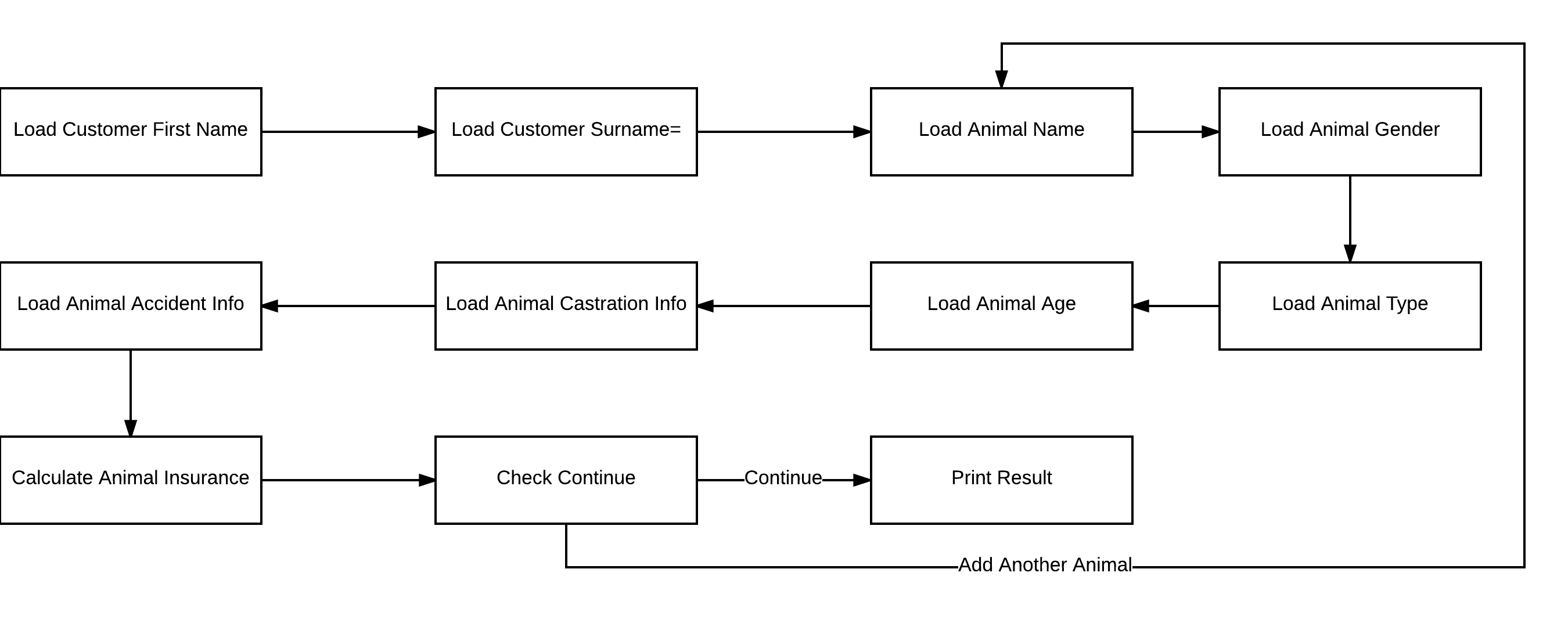
[Check Continue 3](#_Toc481342289)

[Print Result 3](#_Toc481342290)

[Finite-State Machine 3](#_Toc481342291)

# Diagrams

## JSP



## Data Dictionary

### Global

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Load  Pos. | Name | Type | Default | Purpose |
| 1 | curr\_state | int | 0 | Number identifying the current state of program |
| 2 | owner\_first\_name | char array[20] | \0 | The pet owner first name |
| 3 | owner\_surname | char array[20] | \0 | The pet owner surname |
| 4 | pet\_names | char array[10][20] | \0 | The pets names |
| 5 | pet\_gender | int array[10] | -1 | The pets genders |
| 6 | pet\_type | char array[10] | U | The pets types encoded as single characters |
| 7 | pet\_type\_price | float array[5] | 50, 80, 40, 60, 10 | The pets insurance prices |
| 8 | pet\_age | int char[10] | -1 | The pets ages |
| 9 | pet\_is\_neutered | int char[10] | -1 | The pets castration data |
| 10 | pet\_had\_accident | int char[10] | -1 | The pets accident data |
| 11 | insurance\_cost | float char[10] | 0 | The array of insurance total costs |
| 12 | insurance\_base\_price | float char[10] | 0 | The array of insurance base prices |
| 13 | insurance\_mod\_old | float char[10] | 0 | The arrays of insurance modifiers |
| 14 | insurance\_mod\_young\_male | float char[10] | 0 |
| 15 | insurance\_mod\_accident | float char[10] | 0 |
| 16 | quote | float | 0 | The final insurance quote |
| 17 | idx\_curr\_pet | int | 0 | The current index of processed pet |
| 18 | tmp | char array[20] | \0 | Any temporary variable that needs to be stored |

### Local

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Load  Pos. | Name | Type | Default | Purpose |
| 1 | i | int | \0 | Used exclusively within iterative for loops |

# Structures and features

## Loops

I plan to have three loops within my program, two for loops and one while loop. The first for loop will have the purpose of initialising my arrays and setting all values within them to 0 or a null character through \0. The second for loop will be used at the end of the program to echo the arrays back to the user in the form of a final quote with a total. The while loop will serve as a capsule for the state machine to ensure that it keeps running until the final state is reached.

## Conditional Statements

### Switch and Case

I plan to have two switch and case statements. The first of which will be used to handle the type of pet during the calculation stage of the program where the cost of each animal is calculated along with the insurance modifiers. The other switch and cast statement will be used as the main control within the state machine.

### If/Else

If, else and if else statements will be used throughout the program to choose between two or more options where a case statement would not be as necessary or needed at all. The main use for this would be checking if the user has entered a question mark rather than the correct input, as the user requirements state that the program needs to allow the user to use a question mark at any point to get help. The primary difference is that if statements can calculate comparisons, rather than being pre-set.

The program may need to use if statements within case statements if multiple checks are required. Somewhere I will need to use this is when checking if the entered animal is a young male. The program will need to use a case statement to check whether or not the animal is male, and then an if/else statement within to check if the animal is younger than 2 years of age.

#### Validation

The program will also need if/else statements for input validation. If a user enters an invalid input, or something that doesn’t make complete sense in the context, the program needs to be able to recognise this and have the user re-enter the input.

## Finite-State Machine

The core of the program will be controlled by a finite-state machine that has 11 states. Those states are listed in the pseudocode below. The “Check Continue” state has two outcomes. If the user decides to add another animal, it will switch back to the third state, “Load Animal Name”. If they do not, it will continue to the “Print Result” state.

# Pseudocode

## Load Customer First Name

|  |
| --- |
| Print “Customer First Name:”  Scan first\_name  IF first\_name IS ?  Print help  ELSE  nextState |

## Load Customer Surname

|  |
| --- |
| Print “Customer Surname:”  Scan surname  IF surname IS ?  Print help  ELSE  nextState |

## Load Animal Name

|  |
| --- |
| Print “Animal Name:”  Scan animal\_name  IF animal\_name IS ?  Print help  ELSE  nextState |

## Load Animal Gender

|  |
| --- |
| Print “Animal Gender:”  Scan animal\_gender  IF animal\_gender IS ?  Print help  ELSE  nextState |

## Load Animal Type

|  |
| --- |
| Print “Animal Type:”  Scan animal\_type  IF animal\_type IS ?  Print help  ELSE  nextState |

## Load Animal Age

|  |
| --- |
| Print “Animal Age:”  Scan animal\_age  IF animal\_age IS ?  Print help  ELSE  nextState |

## Load Animal Castration Info

|  |
| --- |
| Print “Animal Castration Status:”  Scan animal\_castration  IF animal\_castration IS ?  Print help  ELSE  nextState |

## Load Animal Accident Info

|  |
| --- |
| Print “Animal Accident Status:”  Scan animal\_accident  IF animal\_accident IS ?  {  Print help  }  ELSE  {  nextState  } |

## Calculate Animal Insurance

|  |
| --- |
| Switch animal\_type {  Case Dog  IF animal\_castration IS 1  {  cost = pet\_price[0]  }  ELSE  {  cost = pet\_price[1]  }  Case Cat  IF animal\_castration IS 1  {  cost = pet\_price[3]  }  ELSE  {  cost = pet\_price[4]  }  Case Other  cost = pet\_price[5]  }  IF animal\_age IS GREATER THAN 5  {  cost = mod\_old -> cost  }  ELSE IF animal\_age IS LESS THAN 2 AND animal\_gender IS male  {  cost = mod\_young\_name -> cost  }  IF animal\_accident IS true  {  cost = mod\_accident -> cost  }  nextState |

## Check Continue

|  |
| --- |
| Print “Add Another Animal?”  Scan choice  IF choice IS yes  State0 //Add an animal  ELSE  nextState |

## Print Result

|  |
| --- |
| for ;count IS LESS THAN total\_pets;  {  Print animal\_name  Print cost  Print insurance\_mods  }  Print total  nextState //Ends Program |

## Finite-State Machine

|  |
| --- |
| While state IS NOT 11  {  Switch curr\_state  { case 0:  load\_customer\_first\_name()  case 1  load\_customer\_surname()  Case 2  load\_animal\_name()  Case 3  load\_animal\_gender()  Case 4  load\_animal\_type()  Case 5  load\_animal\_age()  Case 6  load\_animal\_castration\_info()  Case 7  load\_animal\_accident\_info()  Case 8  calculate\_animal\_insurance()  Case 9  check\_continue()  Case 10  print\_result()  }  } |