Final Project – Zoo Monitoring System

Joseph Tooze

SNHU

08/18/2017

Author Note

IT145 – Foundations in Application Development

Final Project – Zoo Monitoring System

# Problem Statement/Scenario

For this project, I will be creating a zoo monitoring system that will monitor various zoo animals and their habitats. The program will ask the user if they want to monitor an animal, a habitat, or to exit the program. Then, it will present the user with a menu and list of animals or habitats to choose from depending on which option the user chooses. A popup dialog will open in the case where the monitor detects something out of the ordinary. The program will also allow the user to go back to the previous options of the menu.

The way I am designing this program is basically, three functions or methods. Each function is wrapped with a WHILE loop that will iterate until the word, “Exit” is entered. When the word “Exit” is entered, the loop will break and go back to the previous menu. The associated “Animals” and “Habitats” files will be read from and parsed. If an alert is detected, a popup box will display showing this alert. Otherwise, it will just display the monitoring information on the console for the user chosen animal or habitat.

## Overall Process

08/15/2017

Upon first starting my code, I took my pseudocode and broke it into 3 different classes. One is the main class where the main menu appears. Another is the class to monitor an animal while the last one is to monitor a habitat.

I struggled a bit with the main menu and method but it wasn’t too bad. I had issues trying to loop unless an exit was called by the user. I did get this loop to work, however. The other part of the main menu that I struggled with was error control. I really don’t want the user to enter any other characters besides for the menu options displayed on the screen. This took quite some thinking, but I finally figured it out and that works now.

I’m now working on the “Monitor Animal” class and it’s going pretty good so far. I was successful in opening the provided file and reading the next submenu. It displays fine on the console and I took the liberty to number the menu items. Now, I just used the same basic principle as my main menu and provided error control. It also goes back to main menu when ‘Q’ or ‘q’ is pressed, which is nice.

08/16/2017

I think I got my “animal” class and associated methods as complete as I can get them. I pretty much just need to use the same basic code for the “habitat” class. I ran into a few problems along the way when it came down to parsing the text file, but I managed to pull it off and it even has error control if you type in the wrong option. I need to make a few changes and add comments in certain areas, but the code works well as a whole.

One of the things I would like to do is make it accept any case when typing in responses. It’s a minor change, but it’s something to think about. I’ve tested and debugged enough for the day and will finish up with the “habitat” part of the program.

08/17/2017

I completed the habitat class today and then I had to do some testing with the final product and fix a few bugs in the program. I had some problems with error control when the user enters an invalid option. I finally got it worked out and retested. Everything works the way it should and I’m trying to finish up commentary and other documentation work for the program.

**Pseudocode**

Function main()

STRING SET userChoice = “”

// First (outer) menu

WHILE userChoice != “Exit”

DISPLAY “Monitor an animal, habitat, or exit program?”

OBTAIN answer from user and SET to userChoice

SWITCH (userChoice)

CASE1: Animal

CALL monitorAnimal()

CASE2: Habitat

CALL monitorHabitat()

DEFAULT:

BREAK

END WHILE

Function monitorAnimal()

STRING SET animalOption = “”

WHILE animalOption != “Exit”

READ options from Animal file

DISPLAY options read from Animal file

DISPLAY “Enter one of the animal options: “

OBTAIN animalOption from user

IF animalOption = “Exit” THEN

BREAK

END IF

READ monitoring information from file depending on animalOption

IF monitoring information contains “\*\*\*\*\*” THEN

OPEN dialog box alerting user that something is out of the ordinary

END IF

ELSE

DISPLAY monitoring information from file

END WHILE

RETURN

Function monitorHabitat()

STRING SET habitatOption = “”

WHILE habitatOption != “Exit”

READ options from Habitat file

DISPLAY options read from Habitat file

DISPLAY “Enter one of the habitat options: “

OBTAIN habitatOption from user

IF habitatOption = “Exit” THEN

BREAK

END IF

READ monitoring information from file depending on habitatOption

IF monitoring information contains “\*\*\*\*\*” THEN

OPEN dialog box alerting user that something is out of the ordinary

END IF

ELSE

DISPLAY monitoring information from file

END WHILE

RETURN

(All programming techniques learned through Zybooks – referenced below)

**Methods and Classes**

The program is broken up into 3 classes. I did not mention these in the pseudocode because I feel pseudocode should be written to be interpreted by any programming language. Java uses classes, but other languages may not. I have the methods in the pseudocode that I used within the program, so no changes were necessary.

ZooMonitor

This class displays the main menu and allows the user to enter one of three options, monitor an animal, monitor a habitat, or exit the program. I have the main method of the program calling other methods from the other two classes. If a user wants to monitor an animal, the program will call the monitorAnimal() method from the MonitorAnimal class. The habitat option works the same way and will call monitorHabitat() from the MonitorHabitat class.

MonitorAnimal

This class displays the next submenu as read from the animals.txt file. It then allows the user to enter one of the animals. For example, a user can enter “Lion” to monitor the lions. The user can also enter ‘q’ to go back to the previous (main) menu and method. The program then parses the txt file for the user chosen animal and displays the monitoring information for that animal. If there is an issue detected (as determined with the \*\*\*\*\* in the text file), the program will call the openDialogBox() method and bring up an information dialog box, displaying a health concern.

MonitorHabitat

This class works the exact same way as the MonitorAnimal class. It gives the user the choice to enter a habitat as read from the habitats.txt file. It also displays the habitat monitoring information. It calls openDialogBox() if a health concern is detected. It also allows the user to go back to the main menu.

**Errors**

I encountered a few major errors that kicked me out of the program. One of those errors was caused by the file and threw an IOException. I did not have the path correct, but quickly resolved this problem. I also encountered several errors such as abnormal behavior after a wrong character or String was entered at the menu prompts. Most of the time, the program would end or I would get caught in an infinite loop. This also happened upon trying to exit the program with ‘q’ or ‘Q’.

**Solutions**

To resolve the IOException, I just corrected the path of the animals.txt or habitats.txt files. To correct the problems with the wrong input being entered, I developed some error control that basically just prints the statement, “Please enter a valid option” and then it redisplays the associated menu. To fix the problem with exiting the program, I had to do some logical thinking about where my While loop should start and end. I found that I had a closing bracket ‘}’ in the wrong place. This caused some unwanted behavior in the program.

References

Zybooks. (2016-2017). SNHU IT145 – Programming in Java. *Zayante.* Retrieved from http://www.zybooks.com