Enhancement One: Software Design and Engineering

Charetta Frierson

Neil Kalinowski

CS 499

22 September 2019

Southern New Hampshire University

Artifact Selection

I have chosen to update and improve the efficiency of the zoo monitoring system from IT 145 Foundations in Application Development. The system helps zookeepers keep track of animal health, activities, and habitats. It uses dialog boxes to alert when an animal or habitat is out of the normal range, displays information based on file, and distinguishes choices based on user selection and category. The artifact was originally created in 2017 and updated September of 2019.

Inclusion and Justification

The artifact was selected to showcase the skills currently asked for within the industry as a Software Developer. The specific components I wanted to reflect was attention to detail, debugging, problem solving, and logical thinking. Overall the program will show software best practices and complete functionality. Initially the program was created in Java, an my idea for the enhancement will be to convert the program to C++ and simpfly the system code to prohibit confusion and lessen complexity.

Outcome and Objectives

Generally, I believe that I did meet the course objectives with the module one enhancement. My goal was to enhance with a different coding language and to simplify so I most definitely did that with the artifact. I do have more updates to make as far as adding the files and making the code the most efficient it can be.

Reflection

Looking back at the enhancement process I would say that it took way more effort than I would have thought to completely change code. Going into the initial enhancement I knew that C++ was unique and found it be a bit less complex than Java. There are some dissimilarities frthe

Java programming language and C++. C++ statements use cout which is a standard output stream. Scope resolution operators are used to define methods outside of a class but uses a local variable with the same name. Unlike Java where the IDE has a built-in debugger, when a programmer uses C++, the programmer must check for their own errors. Lastly structures are supported whereas in Java they are not. Both Java and C++ have their designated pros and cons, and I am ready to face the task. As far as challenges I am still adding the text files, I found a ton of ways to do this and on the next update should have this completed.

Initial Artifact

```
package monitorzoo;
/**
 *
 * @author Charetta Frierson
 */
//Implemented scanners to read .txt files
import java.io.FileReader;
import java.util.HashMap;
import java.util.Map;
import java.io.IOException;
import java.io.BufferedReader;
import java.util.Scanner;

public class MonitorZoo {
  private static Scanner scanner = new Scanner(System.in);
   public static void main(String[] args) {
int choice;
  //Provides operation for the main menu
   while (true) {
   System.out.println("------ Menu ------");
    System.out.println("1. Monitor an animal");
     System.out.println("2. Monitor a habitat");
      System.out.println("3. Exit");
  //Created to handle mutli task options
   choice = scanner.nextInt();
switch (choice) {
    case 1:
     monitorAnimal();
  break;
    case 2:
     monitorHabitat();
  break;
    case 3:
    System.exit(0);//takes user back to main screen

  default:
    System.out.println("Invalid Input");
   }
  }
 }
public static void monitorAnimal() {
  int choice;
   BufferedReader br = null;
    int count = 0;
      Map < Integer, String > animals = new HashMap < Integer, String > ();
System.out.println("------ Animals ------");
  try {
   String sCurrentLine;
    br = new BufferedReader(new FileReader("animals.txt"));//otuputs animal selections
      while ((sCurrentLine = br.readLine()) != null) {
        if (sCurrentLine.startsWith("Details")) {

  String animal = sCurrentLine.split(" ")[2];
    System.out.println(++count + ". " + animal);
      animals.put(count, animal);
     }
    }
   br.close();
   br = new BufferedReader(new FileReader("animals.txt"));
     System.out.println("Enter choice : ");//allows user to select animal by number choice
      choice = scanner.nextInt();
        String selectedAnimal = animals.get(choice);

   if (selectedAnimal != null) {
      while ((sCurrentLine = br.readLine()) != null) {
   if (sCurrentLine.startsWith("Animal") && (sCurrentLine.split(" ")
          [2].equalsIgnoreCase(selectedAnimal) || selectedAnimal.startsWith(sCurrentLine.split("
")[2].toLowerCase()))) {
      for (int i = 0; i < 5; i++) {
   if (sCurrentLine.startsWith("*"))
      System.err.println(sCurrentLine.substring(5));
```

```
       else

      System.out.println(sCurrentLine);
      sCurrentLine = br.readLine();
      }
     }
    }
   }
  } catch (IOException e) {
e.printStackTrace();

  } finally {
   try {
  if (br != null) br.close();
 } catch (IOException ex) {
     ex.printStackTrace();
   }
  }
 }
public static void monitorHabitat() {

  int choice;
   BufferedReader br = null;
    int count = 0;

 Map < Integer, String > habitats = new HashMap < Integer, String > ();
  System.out.println("------ Habitats ------");
   try {
String sCurrentLine;
   br = new BufferedReader(new FileReader("habitat.txt"));//outputs habitat selections
    while ((sCurrentLine = br.readLine()) != null) {
    if (sCurrentLine.startsWith("Details")) {

  String habitat = sCurrentLine.split(" ")[2];
     System.out.println(++count + ". " + habitat);
       habitats.put(count, habitat);
    }
   }
   br.close();
    br = new BufferedReader(new FileReader("habitat.txt"));

   System.out.println("Enter choice : ");//allows user to enter option by number choice
    choice = scanner.nextInt();

   String selectedAnimal = habitats.get(choice);
   if (selectedAnimal != null) {
    while ((sCurrentLine = br.readLine()) != null) {

   if (sCurrentLine.startsWith("Habitat") && (sCurrentLine.split(" ")
            [2].equalsIgnoreCase(selectedAnimal) || selectedAnimal.startsWith(sCurrentLine.split("
")[2].toLowerCase()))) {
     for (int i = 0; i < 4; i++) {
   if (sCurrentLine.startsWith("*"))
        System.err.println(sCurrentLine.substring(5));
   else
       System.out.println(sCurrentLine);
       sCurrentLine = br.readLine();
      }

  } catch (IOException e) {
 e.printStackTrace();

  } finally {
   try {

  if (br != null) br.close();
   } catch (IOException ex) {
    ex.printStackTrace();

   System.exit(0);

   }
 }
```

First Enhancement

```
//ZooMonitor

import <string>;

import <vector>;

import <iostream>;

import <stdexcept>;



namespace ZooMonitor

       export class ZooMonitor
```

```
{
  static void main(std::vector<std::wstring> &args)
  {
       int opt;
       Scanner *sc = new Scanner(System::in);
       ZooMonitor *zm = new ZooMonitor();
       while (true)
       {
 std::wcout << L"Choose the option from the MENU" << std::endl;
  std::wcout << L"------ Menu ------" << std::endl;
  std::wcout << L"1. Monitor an animal" << std::endl;
  std::wcout << L"2. Monitor a habitat" << std::endl;
  std::wcout << L"3. Exit" << std::endl;
  opt = sc->nextInt();
  switch (opt)
  {
  case 1:
  zm->monitorAnimal();
  break;
  case 2:
  zm->monitorHabitat();
  break;
  default:
        std::wcout << L"Invalid Input" << std::endl;

        sc->close();

        break;

        }

                }

                delete zm;

                delete sc;

        }

        // habitat function.

  private:

        void monitorHabitat()

        {

                int opt;

        int num = 0;

        std::wcout << L"------Habitat------" << std::endl;

        File *file = new File(L"habitat.txt");

        Scanner *sc = new Scanner(file);

        while (sc->hasNext())

        {

                std::wstring str = sc->nextLine();

                std::wcout << str << std::endl;

                sc++;

        }
```

```
    sc->close();
        delete sc;
    }
    void monitorAnimal()
    {
        int opt;
    int num = 0;
    std::wcout << L"------Animals------" << std::endl;
    File *file = new File(L"animals.txt");

    Scanner *sc = new Scanner(file);
    while (sc->hasNext())
    {
        std::wstring str = sc->nextLine();
        std::wcout << str << std::endl;
        sc++;
    }
    sc->close();
        delete sc;      }
  }
}
```