CECS 326 – Project 3

"Group Project 3: All About Deadlock"

Due date: 04/04/2023

Students: Alarik Damrow, ID#: 015139330

Pi Oliver, ID#: 02791543

Lab Report: "Group Project 3: All About Deadlock"

1. Goal:

We were provided the following problem description and requirements for this project:

1. Problem Description

Consider that has this kind of scenario: two villages (Eastvillage and Westvillage) only have a single-lane road for the connection. People from these two villages only can use this road for exchange or share their produce. The road can be deadlocked if a people from either East or West on the road simultaneously. To solve this problem to avoid deadlock, please design an algorithm that uses semaphores and/or mutex locks. There have no concerns for the starvation cases.

Implement your solution using synchronization tools. In particular, represent the people at Eastvillage and Westvillage as separate threads (*east_village.java* and *west_village.java*). Once a people is on the road, the associated thread will sleep for a random period of time, representing traveling across the road. You should design a new action for each people when they get into the road, such as eat a donut, to wait for some time. Design your program so that you can create several threads representing the two villages' people without deadlock in the road.

You can flexibly design the algorithm, the test should have no deadlock in the multiple execution cases.

2. Steps:

a. Implement, test, and validate RoadController.java
b. Implement, test, and validate East_village.java
c. Implement, test, and validate West_village.java
d. Develop and provide a README file for the user

We implemented the RoadController project with three Java files: RoadController.java, East_village.java, and West_village.java. The RoadController module creates the road and implements the control flow between the other two modules, East_village and West_village. We used the above steps to develop our program. Code is visible below.

I. RoadController.java

RoadController.java implements the road and handles the control flow for East and West villagers on the road. The program instantiates villagers from each village (East and West) and allows only one villager at a time to travel to the opposite village, avoiding deadlocks with the use of a mutex lock. This program was authored by Alarik Damrow and is the source code is below:

```
/*
This program was created to handle the control of the road and to incorporate the
east and west villagers on such road.
This program will create villagers from both sides and have them go on the road
at certain times to avoid deadlocks.
Author: Alarik Damrow
 */
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;
public class RoadController  // Road to be used
{
    private Lock road = new ReentrantLock(); // Lock needed to run this program
    public void lock() // Lock function for villagers
    {
        road.lock(); // To avoid deadlock
    }
    public void unlock() // Unlock function for villagers
    {
        road.unlock(); // To avoid deadlock
    }
    public static void main (String[] args) // Main to run the whole program with
west and east villagers
    {
        RoadController roadControl = new RoadController(); // One road to be used
        //Set of east villagers
        East_village east1 = new East_village(1,roadControl);
        East_village east2 = new East_village(2,roadControl);
        East_village east3 = new East_village(3,roadControl);
        East_village east4 = new East_village(4,roadControl);
        //Set of west villagers
        West_village west1 = new West_village(1,roadControl);
        West_village west2 = new West_village(2,roadControl);
        West_village west3 = new West_village(3,roadControl);
        West_village west4 = new West_village(4,roadControl);

        //Starting of threads to each villager
```

```
        east1.start();
        east2.start();
        east3.start();
        east4.start();
        west1.start();
        west2.start();
        west3.start();
        west4.start();
    }

}
```

II. East_village.java

East_village.java implements the East village functionality and creates villagers to travel from the East village to the West village. A lock is enabled on the road when the villager is ready to travel, since only one villager may travel at once. Messages are printed to the console when the villager has left the village, performs an actions as they travel, and are done with their travel. This helps with debugging. A random period of sleep between 0-1000ms is forced to try and avoid a deadlock when the villager is done travelling, otherwise the road is unlocked for travel by the next villager. This program is authored by Pi Oliver, and the source code is as follows:

```
/*
This program was created to create east villagers for the road controller that is
to be used.
Author: Oliver Pi
 */
public class East_village extends Thread
{
   private int Villager; // Village number
   private RoadController road; // Road control
   private String[] task = {" is reading astrophysics", " is eating a snadwich
that smacks hard",
   " is planning world domination upon the foos"," is reading Gru's book"}; //
Things that villagers can do
   public East_village (int v, RoadController rc) // Constructor
   {
       Villager = v; // Village number to be passed
       road = rc; // Road to be used
   }
   public void run() // How things are to run
   {
       synchronized (road) // Set the road up
       {
           road.lock(); // Lock that is needed
           System.out.println("East Villager " + Villager + " is on the road");
// On road statement
           System.out.println("East Villager " + Villager +
task[(int)(Math.random()*(4-1+1))]); // Tells user which villager is on the road
           System.out.println("East Villager " + Villager + " has finished his
run"); // Statment saying the end
           try // Force a sleep to avoid possible deadlock
           {
               Thread.sleep((long)(Math.random()*(1000))); // Sleep for any where
from 0 - 1000 ms
           }
```

```
            catch(Exception e) // If error happens
            {

            }
            road.unlock(); // Unlock the thread once job is done
        }
    }
}
```

III. West_village.java

Similarly, West_village.java implements the West village functionality and creates villagers to travel from the West village to the East village. A lock is enabled on the road when the villager is ready to travel, since only one villager may travel at once. Messages are printed to the console when the villager has left the village, performs an actions as they travel, and are done with their travel. This helps with debugging. A random period of sleep between 0-1000ms is forced to try and avoid a deadlock when the villager is done travelling, otherwise the road is unlocked for travel by the next villager. West_village.java operates the same way as East_village.java. This program is authored by Pi Oliver, and the source code is below:

```
private String[] task = {" is listening to Lorna Shore", " is opening the pits",
   " is planning to destroy earth"," is moshing with the homies"}; // Things that
villagers can do
   public West_village (int v, RoadController rc) // Constructor
   {
       Villager = v; // Village number to be passed
       road = rc; // Road to be used
   }
   public void run() // How things are to run
   {
       synchronized (road) // Set the road up
       {
           road.lock(); // Lock that is needed
           System.out.println("West Villager " + Villager + " is on the road");
// On road statement
           System.out.println("West Villager " + Villager +
task[(int)(Math.random()*(4-1+1))]); // Tells user which villager is on the road
           System.out.println("West Villager " + Villager + " has finished his
run"); // Statment saying the end
           try
           {
               Thread.sleep((long)(Math.random()*(1000))); // Sleep for any where
from 0 - 1000 ms
           }
           catch(Exception e) // If error happens
           {

           }
           road.unlock(); // Unlock the thread once job is done
       }
   }
}
```

IV. README

A ReadMe file was developed to aid the user in proper operation of the software program we have developed. The text of the file is provided below and was authored by Alarik Damrow and Pi Oliver.

```
Overall Program:
To run and see the whole function of this program it is best to only run
RoadController.java in either IDE and Terminal. NetBeans IDE was used to write
and run
this program but the program can be run in terminal

IDE Run:
1. Place all java files onto a location that the IDE will be able to recognize
2. Make sure all java files can be loaded by IDE of choice
3. Run the RoadController.java in IDE to see results

Terminal Run:
1. Place all java files onto a location that will be easy to access via the
terminal
2. Open terminal and locate where the java files were placed
3. Run the RoadController.java in terminal with the java command

East_village.java:
This program will create the east villagers needed for the road program
IDE Run:
1. Place East_village.java file onto a location that the IDE will be able to
recognize
2. Make sure East_village.java file can be loaded by IDE of choice
3. Run the East_village.java in IDE to see results
Terminal Run:
1. Place East_village.java file onto a location that will be easy to access via
the terminal
2. Open terminal and locate where the East_village.java file was placed
3. Run the East_village.java in terminal with the java command
*Not Recommended to be Run First. This Program is not the Main Program*

West_village.java:
This program will create the west villagers needed for the road program
IDE Run:
1. Place West_village.java file onto a location that the IDE will be able to
recognize
2. Make sure West_village.java file can be loaded by IDE of choice
3. Run the West_village.java in IDE to see results
Terminal Run:
```

```
1. Place West_village.java file onto a location that will be easy to access via
the terminal
2. Open terminal and locate where the West_village.java file was placed
3. Run the West_village.java in terminal with the java command
*Not Recommended to be Run First. This Program is not the Main Program*

RoadController.java:
This is the main program that will establish the road that is to be used and will
help in the interfacing of the west and east villagers.
Will also run situations between villagers as required.
IDE Run:
1. Place RoadController.java file onto a location that the IDE will be able to
recognize
2. Make sure RoadController.java file can be loaded by IDE of choice
3. Run the RoadController.java in IDE to see results
Terminal Run:
1. Place RoadController.java file onto a location that will be easy to access via
the terminal
2. Open terminal and locate where the RoadController.java file was placed
3. Run the RoadController.java in terminal with the java command
```

3. Results:

RoadController.java, East_village.java, and West_village.java all ran successfully without deadlock. We performed three tests to verify the operation of the program, and we have provided screenshots of the console standard output below to verify the success of the program.

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    Debug: RoadController

(stat_env) C:\Users\Pi\Desktop\CECS 326\326_Project3> c: && cd "c:\Users\Pi\Deskt
cket,server=n,suspend=y,address=localhost:50335 -cp C:\Users\Pi\AppData\Roaming\C
r "
East Villager 1 is on the road
West Villager 4 is moshing with the homies
West Villager 4 has finished his run
West Villager 2 has finished his run
West Villager 1 is on the road
West Villager 1 is listening to Lorna Shore
West Villager 1 has finished his run
East Villager 4 is on the road
East Villager 4 is eating a snadwich that smacks hard
East Villager 4 has finished his run
East Villager 2 is on the road
East Villager 2 is reading astrophysics
East Villager 2 has finished his run
East Villager 3 is on the road
East Villager 3 is planning world domination upon the foos
East Villager 3 has finished his run
```

Figure 1: RoadController.java – Test #1

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

 C:\Users\Pi\AppData\Roaming\Code\User\workspaceStorage\46ec8a6f06ef07d270a9d7e3cd7
ada45\redhat.java\jdt_ws\326_Project3_22d9ddfe\bin RoadController "
East Villager 1 is on the road
East Villager 1 is reading Gru's book
East Villager 1 has finished his run
West Villager 4 is on the road
West Villager 4 is moshing with the homies
West Villager 4 has finished his run
West Villager 3 is on the road
West Villager 3 is moshing with the homies
West Villager 3 has finished his run
West Villager 2 is on the road
West Villager 2 is planning to destroy earth
West Villager 2 has finished his run
West Villager 1 is on the road
West Villager 1 is planning to destroy earth
West Villager 1 has finished his run
East Villager 4 is on the road
East Villager 4 is reading Gru's book
East Villager 4 has finished his run
East Villager 3 is on the road
East Villager 3 is reading astrophysics
East Villager 3 has finished his run
East Villager 2 is on the road
East Villager 2 is reading astrophysics
East Villager 2 has finished his run

(stat_env) C:\Users\Pi\Desktop\CECS 326\326_Project3>
```

Figure 2: RoadController.java – Test #2

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

East Villager 2 has finished his run

(stat_env) C:\Users\Pi\Desktop\CECS 326\326_Project3> c: && cd "c:\Users\Pi\Desktop
\CECS 326\326_Project3" && cmd /C ""C:\Program Files\Java\jre1.8.0_281\bin\java.exe
" -agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:51663 -cp
 C:\Users\Pi\AppData\Roaming\Code\User\workspaceStorage\46ec8a6f06ef07d270a9d7e3cd7
ada45\redhat.java\jdt_ws\326_Project3_22d9ddfe\bin RoadController "
East Villager 1 is on the road
East Villager 1 is reading Gru's book
East Villager 1 has finished his run
West Villager 4 is on the road
West Villager 4 is moshing with the homies
West Villager 4 has finished his run
West Villager 3 is on the road
West Villager 3 is opening the pits
West Villager 3 has finished his run
West Villager 2 is on the road
West Villager 2 is listening to Lorna Shore
West Villager 2 has finished his run
West Villager 1 is on the road
West Villager 1 is planning to destroy earth
West Villager 1 has finished his run
East Villager 4 is on the road
East Villager 4 is reading astrophysics
East Villager 4 has finished his run
East Villager 3 is on the road
East Villager 3 is planning world domination upon the foos
East Villager 3 has finished his run
East Villager 2 is on the road
East Villager 2 is reading Gru's book
East Villager 2 has finished his run

(stat_env) C:\Users\Pi\Desktop\CECS 326\326_Project3>
```

Figure 3: RoadController.java – Test #3

4. Conclusion:

We developed three Java modules to work together to implement the functionality of a road between two villages, East and West that spawn villagers who travel between the villages one at a time along a road. The road utilizes Mutex locks and a random waiting period to ensure no deadlock is possible between the two villages. This program provides an example of synchronization as we learned in the lecture material, and taught us one method of overcoming deadlock in a two process system. We were able to develop and run the program successfully, and learned more about deadlocks and synchronization throughout this lab.