Aaron Turner

Mimi Opkins

4/29/15

Due: 5/6/15

CECS 277

Threads Project

**Project Paragraph**

After racing at least three animals and varying their speed and rest factors, I have determined that rest in the determining factor in an animals placing after the race has finished. I came to this conclusion since, animals with a slightly longer rest period, but much larger speed than other animals tend to place after they do in the race. This may occur since “resting” the animal simply adds more to its overall time in the race, where increasing the speed only slightly shortens the amount of time it take the animal to complete the race. However, if the speed and rest factor is much bigger than another animals speed and rest factor (e.g 800 speed and rest factors vs. 200 speed and rest factors) the animal with the larger speed and rest factor will win. And in this case where speed and rest factors are the same, a speed and rest factor of 500 seems to be an ideal number for an animals to place first in the race.

**AnimalRacer.java**

/\*\*

\* Aaron Turner

\*

\* April 28th, 2015

\*

\* Date Due: May 6th, 2015

\*

\* Purpose: This is the class that extends thread and will act as an "animal" that races

\* other animal racers, and display their progress

\*

\* Input: Parameters to correctly create animal Racers

\*

\* Output: Their position along the race

\*

\*

\*/

**public** **class** AnimalRacer **extends** Thread

{

// name of the animal

**private** String animalName;

// amount of rest an animal will have (milliseconds)

**private** **int** rest;

// the speed of an animal (milliseconds)

**private** **int** speed;

//Animals that have been created

**private** **static** **int** *createdAnimals* = 0;

//Animals that have finished the racd

**private** **static** **int** *finishedAnimals* = 0;

**public** **static** String *winner*;

// our default constructor, create a default animal

**public** AnimalRacer()

{

animalName = "Animal Racer";

rest = 100;

speed = 100;

*createdAnimals*++;

}

// our constructor

**public** AnimalRacer(String name, **int** restInput, **int** speedInput)

{

animalName = "Animal Racer";

rest = restInput;

speed = speedInput;

}

//Set methods

**public** **void** setAnimalName(String name)

{

animalName = name;

}

**public** **void** setSpeed(**int** newSpeed)

{

speed = newSpeed;

}

**public** **void** setRest(**int** newRest)

{

rest = newRest;

}

//Our to string to display our thread info

**public** String toString()

{

//Return our animal

**return** animalName + System.*lineSeparator*() + "Speed: " +

speed + System.*lineSeparator*() + "Rest: " + rest;

}

//Our runnable

**public** **void** run()

{

//save the time we started

Long time = System.*currentTimeMillis*();

//Say that the animal has started

System.*out*.println(animalName + " has Started!");

//Loop to do 5 laps

**for**(**int** i = 1; i < 6; ++i)

{

**try**

{

//race for the lap

**this**.*sleep*(speed);

//Say we completed the lap

System.*out*.println(animalName + " has completed lap # " + i + "!");

//Rest so we may start the next lap

**this**.*sleep*(rest);

}

//If we get an intteruption

**catch** (InterruptedException e)

{

e.printStackTrace();

//inform the user that the animal crash

System.*out*.println("Unfortunately, " + animalName + "Has crashed, and will not race correctly...");

}

}

//increase finished animals

++*finishedAnimals*;

//Prompt finished race

System.*out*.println("Place: " + *finishedAnimals* + ", " + animalName + " has completed the race!");

//If race is finshed tell user program is over

**if**(*createdAnimals* == *finishedAnimals*)

{

System.*out*.println();

System.*out*.println("Race has finished!");

System.*out*.println("Thank you for using the Animal Racer program!");

}

}

}

**MainTest.java**

**import** java.util.ArrayList;

**import** java.util.Scanner;

/\*\*

\* Aaron Turner

\*

\* April 28th, 2015

\*

\* Date Due: May 6th, 2015

\*

\* Purpose: This is supposed to promt the user to create their animal racers,

\* and then race them, and display results to the user

\*

\* Input: Strings and Integers according to the prompts to create and race

\* animals

\*

\* Output: The results of the animals Racing

\*

\*

\*/

**public** **class** MainTest

{

**public** **static** **void** main(String[] args)

{

// Welcome user to the program

System.*out*.println("Welcome to the ANIMAL RACER program!");

// create our scanner to get our input

Scanner input = **new** Scanner(System.*in*);

// Create an arraylist of animal racers

ArrayList<AnimalRacer> animals = **new** ArrayList<AnimalRacer>();

// --- HAVE USER CREATE THE ANIMALS ---

// prompt the user for the number of animals to get

System.*out*.println("How many animals would you like to race? Please input an Integer");

// get the input and make sure it is a number

**int** racers = 0;

String user;

**while** (racers <= 0)

{

user = input.nextLine();

racers = *numCheck*(user);

}

// Now loop until we have created the number of animals

**for** (**int** i = 0; i < racers; ++i)

{

animals.add(**new** AnimalRacer());

}

// Prompt the user to set up every animal

**for** (**int** i = 0; i < racers; ++i)

{

// prompt

System.*out*.println("Please name Animal #" + (i + 1));

// get input and assign to racer

user = input.nextLine();

animals.get(i).setAnimalName(user);

// prompt

System.*out*.println("Give a speed integer (in milliseconds e.g. 100) less than 1000 to Animal #" + (i + 1));

// get the input and make sure it is a number

**int** speed = 0;

**while** (speed <= 0)

{

user = input.nextLine();

speed = *numCheck*(user);

//Since we are subtracting, we need to prompt user for this

**if**(speed >= 1000)

{

System.*out*.println("Please input a number less than 1000...");

speed = 0;

}

}

//Make speed speed - 100 for a better UX

speed = 1000 - speed;

// Add to animal

animals.get(i).setSpeed(speed);

// prompt

System.*out*.println("Give a rest integer (in milliseconds e.g. 100) to Animal #" + (i + 1));

// get the input and make sure it is a number

**int** rest = 0;

**while** (rest <= 0)

{

user = input.nextLine();

rest = *numCheck*(user);

}

// Add to animal

animals.get(i).setRest(rest);

}

// Tell user done racing animals

System.*out*.println();

System.*out*.println("Finished animal racer creation...");

System.*out*.println();

// Finally race the animals, use a boolean for continous races

// Start all the animals

**for** (**int** i = 0; i < animals.size(); ++i)

{

animals.get(i).start();

}

//NOT REQUIRED FOR THE ASSIGNMENT

// boolean racing = true;

// while (racing)

// {

// // Tell the user you are now racing

// // System.out.println("NOW RACING ANIMALS!!!");

//

// // Start all the animals

// for (int i = 0; i < animals.size(); ++i)

// {

// animals.get(i).start();

// }

//

// // Stall output until animals finish

// int finished = 0;

// while (finished < animals.size())

// {

// finished = AnimalRacer.finishedAnimals;

// }

//

// System.out.println();

// System.out.println("Race has finished!");

//

// // Menu to race again

// boolean menu = true;

// while (menu)

// {

//

// System.out.println("Would you like to race again? (Y/N)");

//

// user = input.nextLine();

//

// if (user.equalsIgnoreCase("Y"))

// {

// //They want to race again

// menu = false;

// }

// else if (user.equalsIgnoreCase("N"))

// {

// //They do not want to race again

// racing = false;

// menu = false;

// }

// else

// {

// // it was improper input

// System.out.println("That is not valid input, please enter Y or N");

// }

// }

// }

// leave user from the program done in animal racer

}

// Function to determine if input is a number

**public** **static** **int** numCheck(String number)

{

// try and parse the int, if it is not an int catch and return

// false

**int** returner = 0;

**try**

{

returner = Integer.*parseInt*(number);

**if** (returner < 0)

System.*out*.println("That is not positive, Please input a positive Integer");

}

**catch** (NumberFormatException nfe)

{

System.*out*.println("That is not an integer, Please input an Integer");

returner = 0;

}

finally

{

return returner;

}

}

}

**Sample Output**

Welcome to the ANIMAL RACER program!

How many animals would you like to race? Please input an Integer

3

Please name Animal #1

Cheetah

Give a speed integer (in milliseconds e.g. 100) less than 1000 to Animal #1

900

Give a rest integer (in milliseconds e.g. 100) to Animal #1

400

Please name Animal #2

Grizzly Bear

Give a speed integer (in milliseconds e.g. 100) less than 1000 to Animal #2

750

Give a rest integer (in milliseconds e.g. 100) to Animal #2

600

Please name Animal #3

Mountain Goat

Give a speed integer (in milliseconds e.g. 100) less than 1000 to Animal #3

350

Give a rest integer (in milliseconds e.g. 100) to Animal #3

100

Finished animal racer creation...

Cheetah has Started!

Grizzly Bear has Started!

Mountain Goat has Started!

Cheetah has completed lap # 1!

Grizzly Bear has completed lap # 1!

Cheetah has completed lap # 2!

Mountain Goat has completed lap # 1!

Grizzly Bear has completed lap # 2!

Cheetah has completed lap # 3!

Mountain Goat has completed lap # 2!

Cheetah has completed lap # 4!

Grizzly Bear has completed lap # 3!

Cheetah has completed lap # 5!

Mountain Goat has completed lap # 3!

Place: 1, Cheetah has completed the race!

Grizzly Bear has completed lap # 4!

Mountain Goat has completed lap # 4!

Grizzly Bear has completed lap # 5!

Mountain Goat has completed lap # 5!

Place: 2, Mountain Goat has completed the race!

Place: 3, Grizzly Bear has completed the race!

Race has finished!

Thank you for using the Animal Racer program!