Final Project: Program Specification- PollutionCombat

Shakhzod Ali-Zade && Dennis Mbae && Dickson Wafula

Part 1

What will your program look like at the end?

We are creating a simulation which takes place underwater. Therefore, the background will be blue. In addition, we will have several game objects: tank, aircraft, submarines, ships, clouds and fish. User will manipulate tank to destroy other objects and reduce the pollution. The objects will be dropping toxins into the ocean. Aircrafts will be moving on the top, ships will be on the top of the ocean and submarines under the ocean. The game is over once the toxins in the sea reach a certain level to be determined as we continue in the development of this program.

How will the user input work? (e.g., clicking, keys on the keyboard, specifying a file, what have you)

The user will be a player pressing on the keyboard to control the tank which is used to eliminate the environmental pollutants. The tank can only move left and right and its cannon will be able to shoot at certain angles. The program will respond by shooting bullets from the canon in the tank to whatever enemy they are pointing to.and eliminating them from the GUI when they become dead and advancing to higher levels as the game proceeds.

How will the program respond?

The program will respond by assigning the gameObjects certain roles as the game proceeds. The villains will drop waste in the ocean as we the player tries to eliminate them.

What purpose does it serve? (e.g., is a game, a productivity tool, a screen saver?)

This program will be a program that serves as a game and a tool to demonstrate the environmental impact of human activities in the ocean.

Part 2

It was hard for us to describe some of the aspects of our program without sharing some snippets of code. We have provided snippets of code along some of our explanations.

PollutionCombat.java

Variables in PollutionCombat.java

-Width of the screen

private static final int WIDTH=1024;

-Height of the screen

private static final int HEIGHT=768;

-Number of frames per second

private static final int FPS = 60; //Frames per Second

-The main Thread

private Thread thread; //is the mainThread

-The World

private World world; //is the world that contains all our game objects

-Whether the thread is running or not.

private boolean isRunning=false; //whether the thread is running or not.

The main class is class PollutionCombat which extends JPanel and implements Runnable. In this class, the main method sets up the GUI and creates a new PollutionCombat.

The pollution combat constructor creates a new World class and adds a KeyListener. It the calls on the method start() where the mainThread of the program is started. This goes to the @ override run() method where our program does the actual work of updating and repainting the GUI using the update() and repaint() methods. The pollutionCombat is created only once.

```
public PollutionCombat(){ //constructor

   this.setPreferredSize(new Dimension(WIDTH, HEIGHT)); //display size

   world=new World(); //create a new world--the world is almost like a level

   this.addKeyListener(new KeyInput(world)); //create a new Listener for our
world

   start();//start the Program

}
```

World.java

This class contains the objects in our game.

Variables in World.java

-All the objects in our game are contained in a Vector

```
Vector<GameObject> gameObjects;//all the gameObjects in our world
```

Methods in World.java

-Repainting method- repaints the objects in the game after every update

repaint() //repaints

-Updating method- updates the objects in our LinkedList

update() //updates

- AddObject(GameObject)- Adds gameObject to LinkedList gameObjects

-Remove object()-Removes gameObject from LinkedList gameObjects

Constructor

The constructor just creates an empty LinkedList of gameObjects

```
public World(){//what will a typical world have?
   gameObjects=new Vector<>();//all the gameObjects
}
```

Abstract Class GameObject

This is the backbone for the characters in our game

Each object in the game must follow the following rules as indicated by the following variables and methods

Every object must:

-Must have a position x and y

```
protected int x,y; //position on the screen
```

-Must have a motion factor

```
protected float velX=0, velY=0; //defines the motion
```

-Must have an ID

```
protected Character ID;
```

-Must draws something new after 60 FPS

```
public abstract void repaint(Graphics g);//draws everything(Graphics g);
//every object
```

-Must update

```
public abstract void update(); //every object updates
```

-When we construct a gameObject the constructor assigns to it an ID, position x and y.

```
public GameObject(int posX,int posY , Character ID){ //velocity

   this.x=posX; this.y=posY;  this.ID=ID;

}
```

ENUM CHARACTER

Contains the various IDs for our gameObjects in the game

TANK; which represents the shooter in the sea

BULLET; which contains the bullets tank uses

SUBMARINE; which represents a nuclear submarine

SHIP; which represents the ships with toxins

AEROPLANE; which represents the aeroplanes

HOW CLASSES IN OUR PROGRAM INTERACT

The following is a diagram of how the classes interact: