CSC110AB
Program4

The problem:

You will have to create a "Tank" and guide it through a "Landscape" in order to move around a simple obstacle course to reach a "Target." Everything you need will be provided to you in predefined classes; you have to use the classes and your own logic to tell the Tank how to reach the Target.

Classes you will use:

To use each of these classes, you will get an instance of the class (either created by you or returned from a method) and then tell it to do things (its methods).

The following classes are built into Java. We have previously used some of them. They are all documented on the java.sun.com website under "APIs". The package they are in is shown at the very top of the documentation; if the class is in a specific package (other than java.lang), be sure and import that package.

- String — We have used this class already.
- Scanner — Can be used to get keyboard input (we have done this already)
- Random — Used to create a random number generator. After you create one, you can tell it to return its nextInt(<num>), which will return a random number between 0 and <num> -1. (we have used this already, but you can use Math.random() instead if you'd rather).
- Point — Represents an (x,y) coordinate. Note that when you tell a Point to getX(), it will return the value as a double. Since we are working with pixels on a screen (integers), you will have to typecast it to an int.
- Color — Represents a color. It stores each of the RGB (Red, Green, Blue) components of the color as an integer from 0-255. So the statement:

```
Color myColor = new Color(255, 0, 255);
```

would create a Color that has as much red as possible, no green, and as much blue as possible. The result would be purple. The Color class also has some "static" values. For example, the statement:

```
Color myColor = Color.BLUE;
```

would create a Color that is blue. See the java documentation for the predefined Colors.

You are also given 2 classes that I wrote. The classes are:

- Tank — Describes a drawable "Tank" (which is drawn as a square). You are not to change its code. It has the following constructors and methods (if you look at Tank.java, you may see a few others that you can use if you want, but the ones below are the ones you will need). Note that the following show how they are defined (like the listings in the java.sun.com classes), not the exact way they would be called…

```
//default constructor - sets it to Black and VERY slow
public Tank()

//"parameterized constructor" - receives a color and a speed (1-30)
public Tank(Color theColor, int theSpeed)
```

```
//"parameterized constructor" - receives a color and a speed (1-30) and leaves a "trail"
//       so you can see its route (if last argument is true).
public Tank(Color theColor, int theSpeed, boolean leavesTrail)

// move - changes its position by 1 pixel
public void move()

// getDimension - returns the dimension (both length and width, since its a square)
public int getDimension()

// getPositionX() - returns its x coordinate
public int getPositionX()

// getPositionY() - returns its y coordinate
public int getPositionY()

// setPosition – sets its position to whatever x and y are passed in
public void setPosition(int x, int y)      (you are not allowed to use this method)

// setColor - sets its color to whatever is passed in
public void setColor(Color newColor)

// turn - "turns" by changing its orientation
public void turn(String whichWay)      (you can pass in "left" or "right")

// reverse - puts it in "reverse gear" so it goes in reverse
public void reverse()

// forward - turns off reverse gear so it moves forward
public void forward()

//toString - returns its representation as a String
public String toString()
```

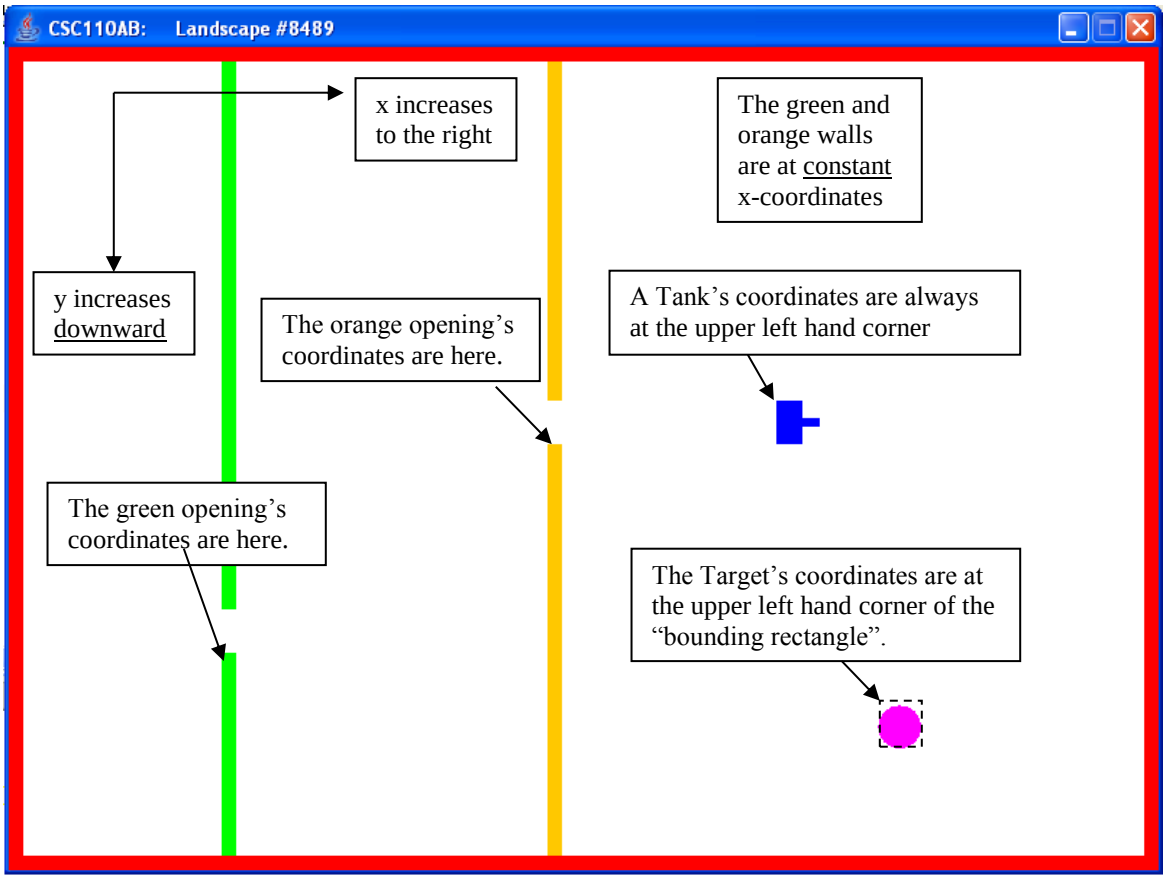
- Landscape — Describes a drawable "Landscape" which is a simple obstacle course consisting of 2 Walls (with an opening that can fit a Tank) and a "Target." You are not to change its code.

Landscape's constructors and methods are listed below. Note that the following show how they are defined (like the listings in the java.sun.com classes), not the exact way they would be called…

```
// Default constructor - generates a random position for the green opening, the orange
//                       opening, and the target.
public Landscape()

// Parameterized constructor which receives an int that is the landscapeID
public Landscape(int theID)          (must be between 0000 and 9999)

// getOrangeOpening() - returns the opening in the orange wall as a Point (the top/left
//                      point of the lower section)
public Point getOrangeOpening()

// getGreenOpening() - returns the opening in the green wall as a Point (the top/left
//                     point of the lower section)
public int getGreenOpening()
```

```
// getTargetLocation - returns a Point which is the location of the target (actually the
//                     upper/left point of its bounding rectangle)
public Point getTargetLocation()

// addTank - adds the Tank that is received to this instance of Landscape
public void addTank(Tank t)
```

An example of a Landscape instance, along with some relevant notes, is shown below:



Your program:

Your program should be named Program4.java. It will contain the main method; inside main should be code to do the following:

- Prompt the user to enter their name then use the Scanner class to read it in

- If their name is "Randy" (don't use == for Strings; use .equalsIgnoreCase here), then
  - Create a new Color that consists of a random amount of Red, a random amount of Yellow, and a Random amount of Blue (different random numbers).
  - Create a new instance of a Tank, passing in the Color you created and any speed you want.
- But if their name is "Smurfette" (don't use == for Strings; use .equalsIgnoreCase here), then
  - Create a new instance of a Tank, passing in some shade of Blue and any speed you want. The shade of Blue can be either the "static" value you get from the class (see my brief

description of the Color class above), or a custom Color you created with no Red, no Green, and some amount of Blue.

- Otherwise, create a new instance of a Tank with whatever non-Blue Color you want (see my brief description of the Color class above) and any speed you want.

- Once your Tank is created, you should create a new instance of a Landscape. You should call its default constructor, which will create one with random openings in the walls and target location. The ID for that particular Landscape will be displayed at the top.
  - NOTE: for testing purposes, you can create a new instance of a Landscape by passing the ID into the constructor. That way, you test your logic against a particular Landscape over and over as you debug it. Your final version of Program4 should call its default constructor so that it creates a random Landscape.

- Tell your Landscape to add your Tank to itself.

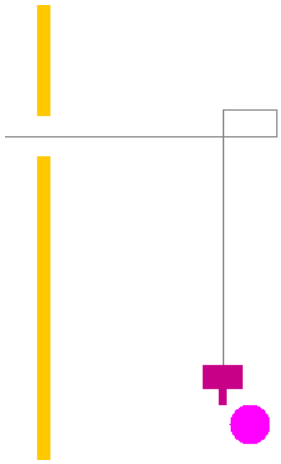
- Tell your Landscape and Tank to return whatever unchanging data (location of openings, dimension of the Tank, location of the target) you will need to program the commands that tell the Tank to successfully move through the Landscape.
  - Note: when you ask the Landscape to getGreenOpening(), it will return it as a Point. What you will eventually need is the x and y values, so be sure and get those values out of the Point that is returned. The same is true for other methods that return a Point.

- Once you have all the unchanging data, figure out and program the logic to tell your Tank to move through the openings and to the target.
  - You can only move() one pixel at a time, so if you want to move any distance you should put the move() statement in a loop.
  - The location of the Tank will change as it moves. To keep your calculations up to date, you will have to ask the Tank over and over what its position is.

Additional requirements: As your program tells the Tank to move through the obstacle course, it should also do the following:

1. Tell the tank to change to reverse gear at least twice during its journey. After you do this, it's moves will be backwards (in reverse). To make it move forwards again, tell it to do its forward() method. See Tank's list of methods for details.

2. Your tank's path should cross itself at least once. If you use the constructor that shows the "trail," you will see the path it took. It should do something like this (anywhere in the path you want):



3. Your tank should change color (noticeably) the first time it turns after passing through the Green wall.

Testing your program: Please note:

1. Every time a Landscape is created, its 4-digit ID number will be displayed in the title bar. Landscape has a constructor that will allow you to pass in a 4-digit ID number if you want to keep testing with the same Landscape.
2. If you cannot figure out how to get it to work for any Landscape, you can get a small amount of partial credit for making it work for #6147 and way you can ("hard-coding" the Tank's movements).
3. If you put System.out.println's in your code for debugging, they will print in the black DOS window behind the Landscape frame.

Grading: The following Rubric shows how your program will be graded for functionality. Deductions for not following requirements, lateness, maintainability (commenting, variable names, indenting, etc) will be taken off in addition.

a Landscape has been created	4 pts.	
a Tank has been added to the Landscape	4 pts.	
if the user entered "Randy" (or "rANdY"), the Tank is a random Color	8 pts.	
if the user entered "Smurfette" (or "sMurFEtTe"), the Tank is some shade of Blue	4 pts.	
if the user entered neither of the above names, the Tank is some non-Blue Color	4 pts.	
the Tank turns around from starting position	4 pts.	
the Tank visibly moves from starting position (more than just a pixel or two)	6 pts.	
the Tank is told to change to reverse gear at least twice	2 pts.	
the Tank's path crosses back over itself at least once	2 pts.	
the Tank changes color the first time it turns after passing the Green wall.	2 pts.	
the Tank can solve any dynamic (random) Landscape		
successfully goes through Green Wall	8 pts.	
then turns in the correct direction eventually	8 pts.	
successfully goes through Orange Wall	8 pts.	
then turns toward Target eventually	8 pts.	
runs into Target	8 pts.	

Please submit: Your Program4.java via Canvas. I will already have Tank.java and Landscape.java.