

Name : _____

Seat No : _____

Paper IV(Robotics and Artificial Intelligence)

INDEX				
NO	DATE	TITLE	PAGE NO	SIGN
1		Write a program to create a robot (i) With gear (ii) Without gear and move it forward, left, right		
2		Write a program to create a robot with a two motor and move it forward, left, right		
3		Write a program to do a square using a while loop, doing steps with a for loop, to change directions based on condition, controlling motor speed using switch case,		
4		Write a program to create a robot with light sensors to follow a line		
5		Write a program to create a robot that does a circle using 2 motors		
6		Write a program to create a path following robot		
7		Write a program to resist obstacles		
8		Write a program to implement Torch following robot		

Practical no. 1(a)

```
import ch.aplu.robotsim.*;
class MoveWithGear
{
    MoveWithGear()
    {
        NxtRobot robot=new NxtRobot();
        Gear gear=new Gear();
        robot.addPart(gear);

        gear.forward(400);
        gear.setSpeed(30);

        gear.left(800);
        gear.forward(200);
        gear.right(480);
        robot.exit();
    }
    public static void main(String args[])
    {
        MoveWithGear m=new MoveWithGear();
    }
}
```

Output:



Practical no. 1(b)

```
import ch.aplu.robotsim.*;
class MoveWithoutGears
{
    MoveWithoutGears()
    {
        TurtleRobot robot=new TurtleRobot();
        robot.forward(100);
        robot.left(45);
        robot.forward(200);
        robot.right(90);
        robot.backward(100);
        robot.exit();
    }
    public static void main(String args[])
    {
        MoveWithoutGears m=new MoveWithoutGears();
    }
}
```

Output :



Practical No. 2

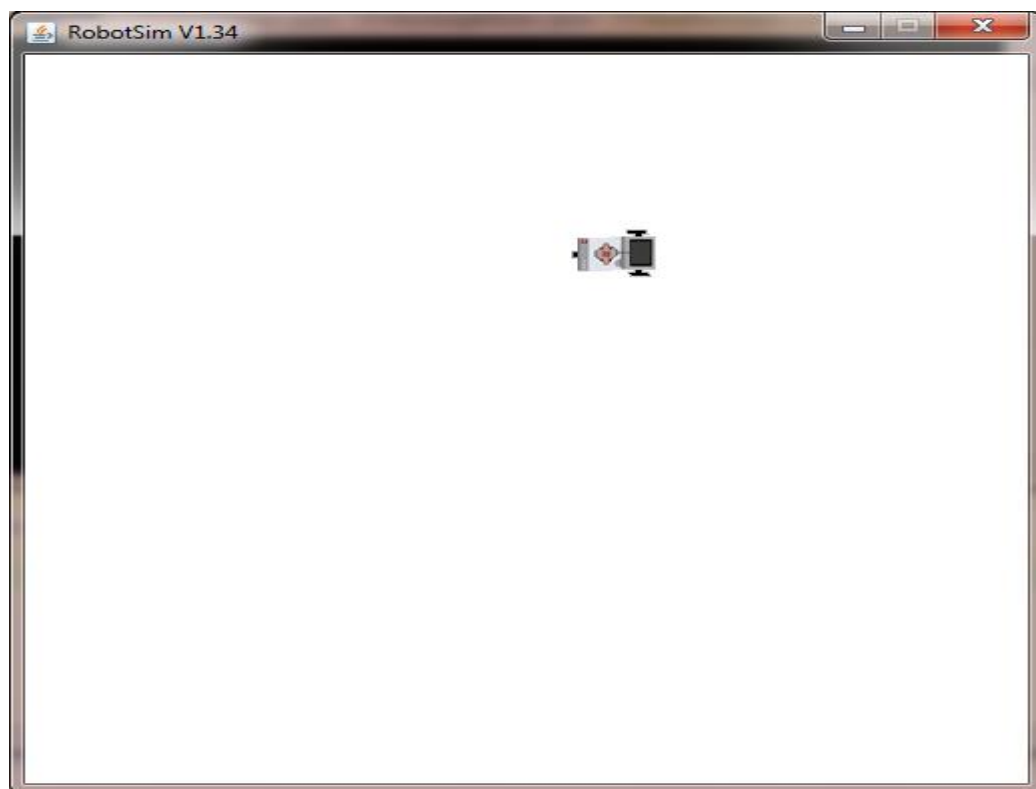
```
import ch.aplu.robotsim.*;
public class MoveWithMotors
{
    public MoveWithMotors()
    {
        NxtRobot robot=new NxtRobot();
        Motor motA=new Motor(MotorPort.A);
        Motor motB=new Motor(MotorPort.B);
        robot.addPart(motA);
        robot.addPart(motB);
        motA.forward();
        motB.forward();
        Tools.delay(2000);

        motA.stop();
        Tools.delay(1050);
        motA.forward();
        Tools.delay(2000);

        motB.stop();
        Tools.delay(1050);
        motB.forward();
        Tools.delay(2000);

        robot.exit();
    }
    public static void main(String args[])
    {
        new MoveWithMotors();
    }
}
```

Output:



Practical no. 3

```
import ch.aplu.robotsim.*;
class square
{
    square()
    {
        NxtRobot r = new NxtRobot();
        Gear g = new Gear();
        r.addPart(g);
        g.setSpeed(100);
        g.forward(1000);
        g.left(275);
        g.forward(1000);
        g.left(275);
        g.forward(1000);
        g.left(275);
        g.forward(1000);

        Tools.delay(2000);
        r.exit();
    }
    public static void main(String[] args)
    {
        new square();
    }
}
```

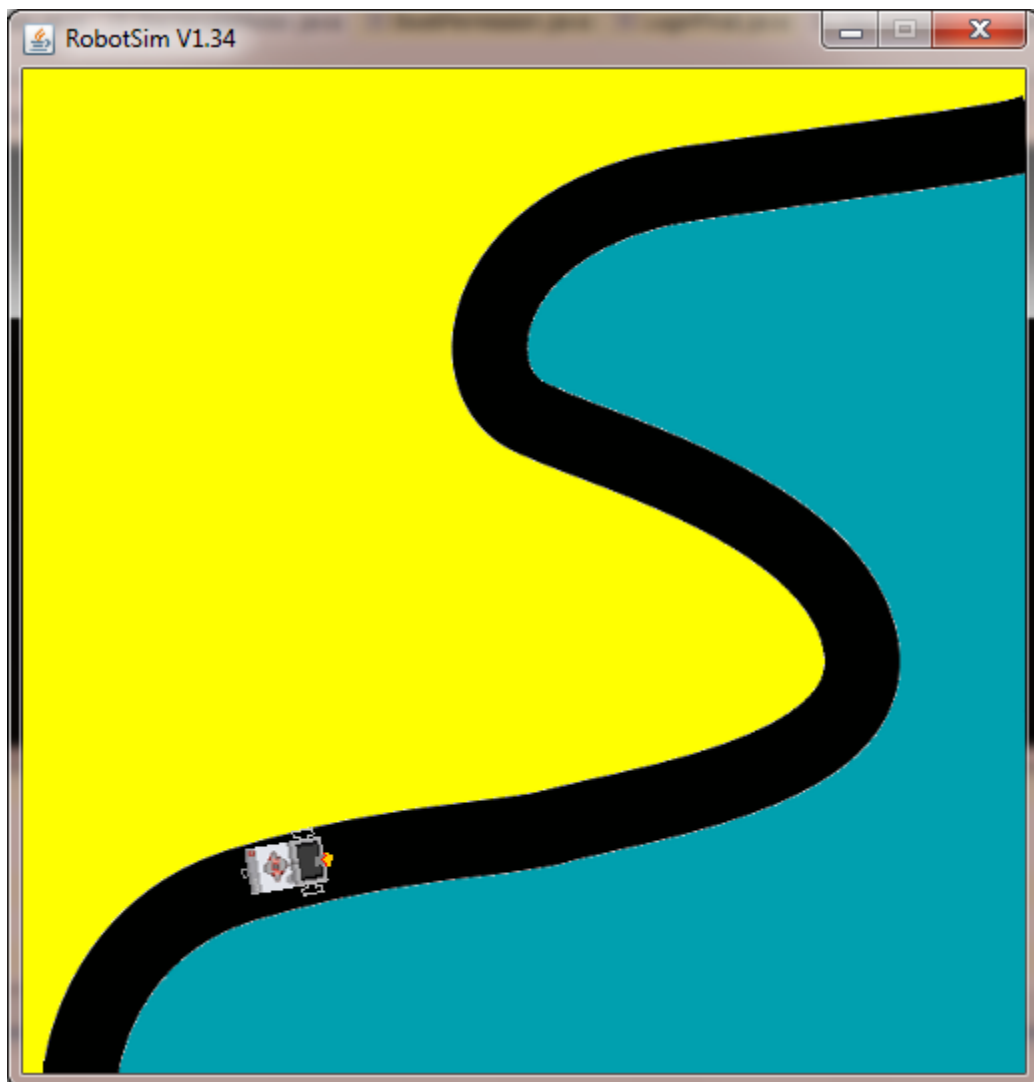
Output:



Practical no. 4

```
import ch.aplu.robotsim.*;
public class LineFollower
{
    LineFollower()
    {
        LegoRobot robot=new LegoRobot();
        Gear gear=new Gear();
        LightSensor ls=new LightSensor(SensorPort.S3);
        robot.addPart(gear);
        gear.setSpeed(20);
        robot.addPart(ls);
        while(true)
        {
            int v=ls.getValue();
            if(v < 100)//black
                gear.forward();
            if(v > 300 && v < 750) //blue
                gear.leftArc(0.05);
            if(v> 800) //yellow
                gear.rightArc(0.05);
        }
    }
    public static void main(String args[])
    {
        new LineFollower();
    }
    static
    {
        RobotContext.setStartPosition(50,490);
        RobotContext.setStartDirection(-90);
        RobotContext.useBackground("sprites/road.gif");
    }
}
```

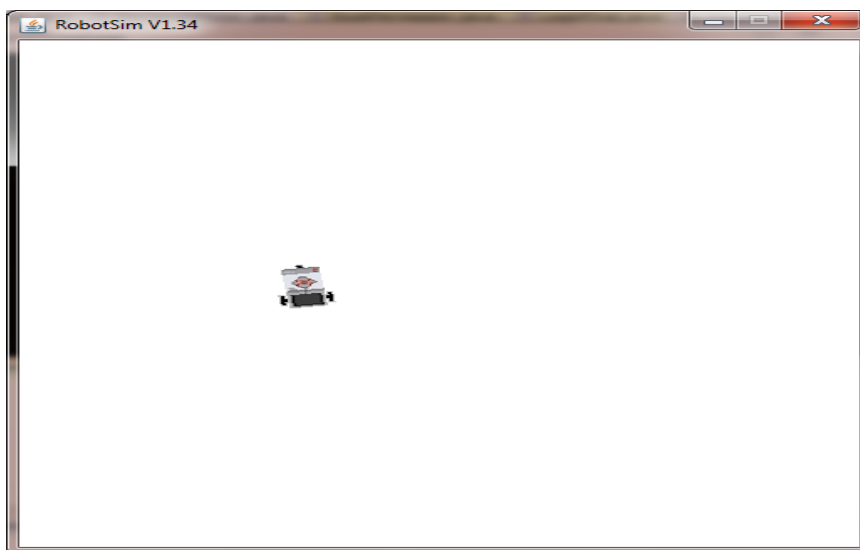
Output:



Practical No. 5(a)

```
import ch.aplu.robotsim.*;
public class Circlem
{
    Circlem()
    {
        NxtRobot robot=new NxtRobot();
        Gear gear=new Gear();
        robot.addPart(gear);
        gear.setSpeed(60);
        gear.leftArc(0.2,7000);
        gear.rightArc(0.2);
        Tools.delay(5000);
        robot.exit();
    }
    public static void main(String args[])
    {
        new Circlem();
    }
}
```

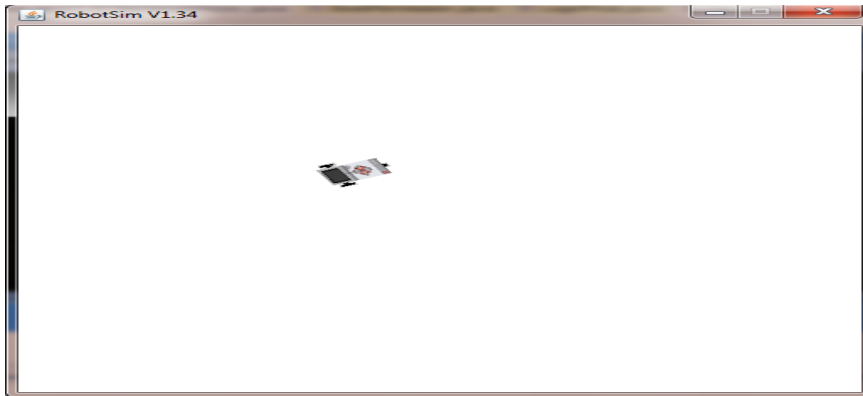
Output:



Practical No. 5(b)

```
import ch.aplu.robotsim.*;
class CircularGear
{
    CircularGear()
    {
        NxtRobot robot=new NxtRobot();
        Gear gear=new Gear();
        robot.addPart(gear);
        gear.forward(200);
        gear.setSpeed(20);
        gear.leftArc(0.2,7000);
        gear.forward(200);
        gear.leftArc(0.2,7000);
        gear.forward(200);
        gear.leftArc(0.2,7000);
        gear.forward(200);
        gear.leftArc(0.2,7000);
        gear.forward(200);
        robot.exit();
    }
    public static void main(String args[])
    {
        CircularGear m=new CircularGear();
        NxtContext.setStartPosition(250,200);
        NxtContext.setStartDirection(90);
    }
}
```

Output:



Practical No. 6

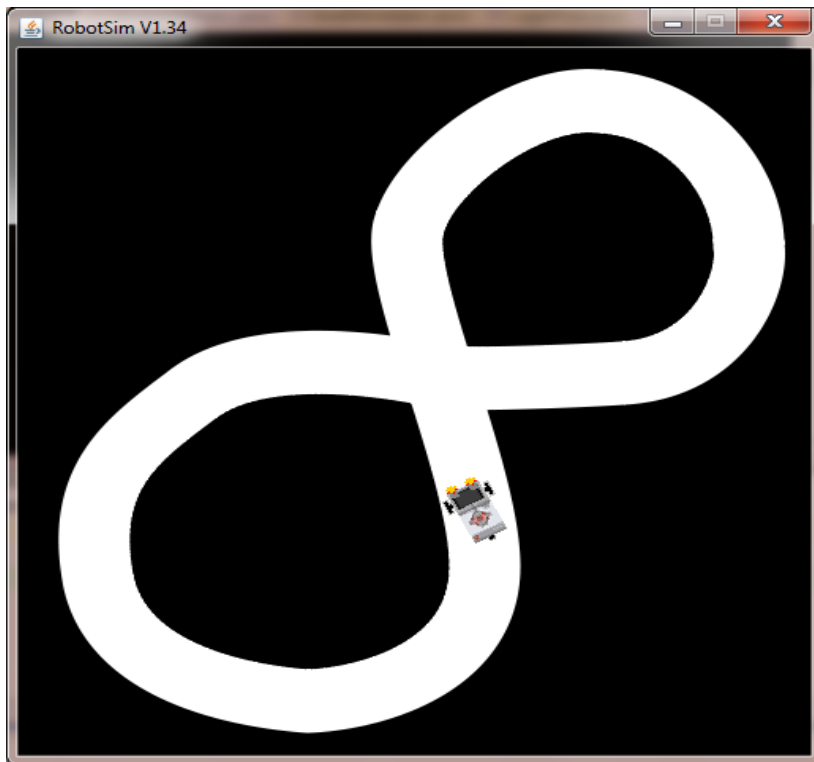
```
import ch.aplu.robotsim.*;
public class Pathfinder
{
    public Pathfinder()
    {
        NxtRobot robot=new NxtRobot();
        Gear gear=new Gear();
        LightSensor ls1=new LightSensor(SensorPort.S1);
        LightSensor ls2=new LightSensor(SensorPort.S2);
        robot.addPart(gear);
        robot.addPart(ls1);
        robot.addPart(ls2);
        gear.forward();

        while(true)
        {
            int rightValue=ls1.getValue();
            int leftValue=ls2.getValue();
            int d=rightValue - leftValue;
            if(d>100)
                gear.rightArc(0.1);
            if(d < -100)
                gear.leftArc(0.1);
            if(d > -100 && d < 100 && rightValue > 500)
                gear.forward();
        }
    }
    public static void main(String args[])
    {

```

```
    new Pathfinder();  
}  
static  
{  
    NxtContext.setStartPosition(250,490);  
    NxtContext.setStartDirection(-90);  
    NxtContext.useBackground("sprites/path.gif");  
}  
}
```

Output:



Practical no. 7

```
import ch.aplu.robotsim.*;
import ch.aplu.util.*;
public class resistobst
{
    public resistobst()
    {
        LegoRobot robot = new LegoRobot();
        Gear g = new Gear();

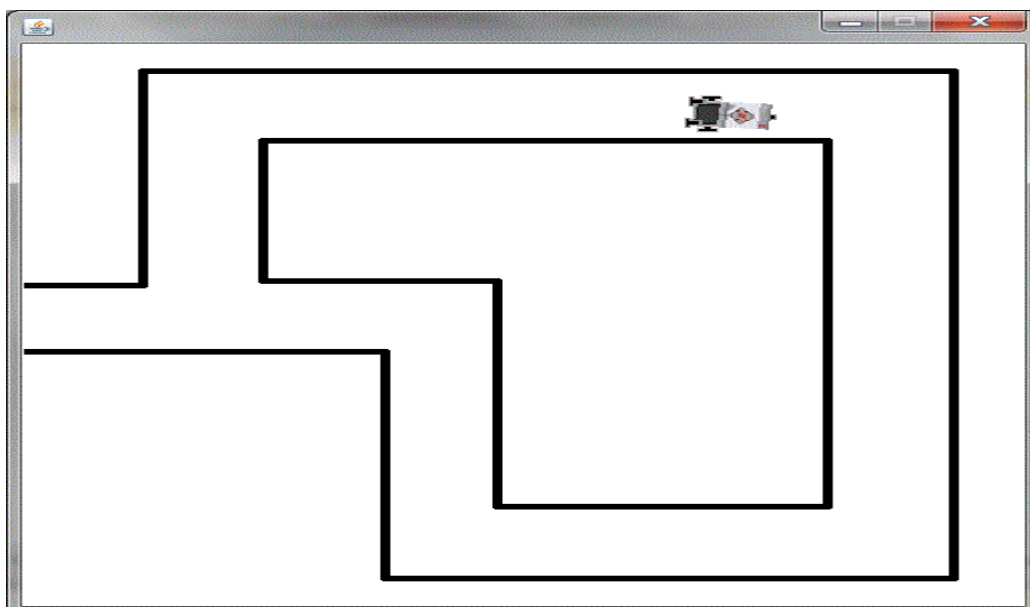
        TouchSensor ts1 = new TouchSensor(SensorPort.S1);
        TouchSensor ts2 = new TouchSensor(SensorPort.S2);
        robot.addPart(g);
        robot.addPart(ts1);
        robot.addPart(ts2);
        g.forward();
        while(!QuitPane.quit())
        {
            Boolean t1 = ts1.isPressed();
            Boolean t2 = ts2.isPressed();
            if(t1 && t2)
            {
                g.backward(500);
                g.left(400);
                g.forward();
            }
            else
            {
                if(t1)
                {
                    g.backward(500);
                    g.left(400);
                    g.forward();
                }
                else
                {
                    if(t2)
```

```

        {
            g.backward(500);
            g.right(100);
            g.forward();
        }
    }
}
Tools.delay(20);
}
robot.exit();
}
public static void main(String [] args)
{
    new resistobst();
}
static
{
    RobotContext.setLocation(10,10);
    RobotContext.setStartDirection(5);
    RobotContext.setStartPosition(100,240);
    RobotContext.useObstacle(RobotContext.channel);
}
}

```

Output:



Practical no. 8

```
package TorchFollower;

import ch.aplu.robotsim.Gear;
import ch.aplu.robotsim.LegoRobot;
import ch.aplu.robotsim.LightSensor;
import ch.aplu.robotsim.RobotContext;
import ch.aplu.robotsim.SensorPort;
import ch.aplu.robotsim.Tools;

/**
 *
 */
public class TorchFollower {
    TorchFollower()
    {
        LegoRobot robot = new LegoRobot();
        LightSensor lsFR = new LightSensor(SensorPort.S1, true);
        LightSensor lsFL = new LightSensor(SensorPort.S2, true);
        LightSensor lsRR = new LightSensor(SensorPort.S3, true);
        LightSensor lsRL = new LightSensor(SensorPort.S4, true);

        Gear gear = new Gear();
        robot.addPart(gear);
        robot.addPart(lsFR);
        robot.addPart(lsFL);
        robot.addPart(lsRL);
        robot.addPart(lsRR);

        gear.setSpeed(25);
        gear.forward();
        double s = 0.02;
        while (!robot.isEscapeHit())
        {
            int vFR = lsFR.getValue();
            int vFL = lsFL.getValue();
            int vRR = lsRR.getValue();
```

```

int vRL = lsRL.getValue();
double d = 1.0 * (vFL - vFR) / (vFL + vFR);

if (vRL + vRR > vFL + vFR) // torch behind robot
    gear.left();
else if (d > -s && d < s)
    gear.forward();
else
{
    if (d >= s)
        gear.leftArc(0.05);
    else
        gear.rightArc(0.05);
}
Tools.delay(100);
}
robot.exit();
}

public static void main(String[] args)
{
    TorchFollower t = new TorchFollower();
}
// ----- Environment -----
static
{
    RobotContext.useTorch(1, 150, 250, 100);
}
}

```


Output

