Escola Superior de Tecnologia e Gestão Instituto Politécnico da Guarda	**Laboratorial Work**

Nº 3				
Group: 31				
	Name	**Ednilson Wagner**	*Nº*	**1011276**
	Name	**Janilta Pires**	*Nº*	**1012063**
	Name	**Janilza Simão**	*Nº*	**1012414**

Fill in the header record with the name and number of the group members and add "(missing)" in case any member of the group has missed the class. After finishing the laboratorial work, add it to the group files section in Blackboard.

1. Program the robot to traverse a straight distance as close as possible to 50 cm. Test the program on different surfaces.

Program:

```
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.ContinuousRotationServo;
import com.ridgesoft.robotics.Motor;

public class ManeuverWithServo {
    private static Motor leftMotor;
    private static Motor rightMotor;
   public static void main(String args[]) {
            System.out.println("Maneuver I");
            leftMotor =  new ContinuousRotationServo(IntelliBrain.getServo(1), false);
            rightMotor = new ContinuousRotationServo(IntelliBrain.getServo(2), true);

            //Andar para a frente
            leftMotor.setPower(8);
            rightMotor.setPower(8);
            try {
                Thread.sleep(2500); //mais ou menos 60cm
```

```
            } catch (InterruptedException e){}
            leftMotor.stop();
            rightMotor.stop();
    }
}
```

The robot ran the same distance on different surfaces? Why?

R: Consoante o tipo de superficie ele pode ter mais ou menos atrito, por isso a distância percorrida é diferente.

2. Create a method (function) to make the robot rotate the number of degrees passed as a parameter: turnAngle(int angle).

   Tip: Create a formula to convert the value of the angle passed as a parameter to the value of time the robot has to rotate. Adjust the formula for a given type of surface.

   Formula: time = angle*(300/90) porque para girar 90 graus o robô gira durante 300ms a velocidade 8 como vimos na função rotatte90 do tutorial.

   Program:

```
public static void turnAngle(int angle){
            final double CONST= 300/90;
            double time = angle * CONST;

            leftMotor.setPower(8);
            rightMotor.setPower(-8);
            try {
                  Thread.sleep((int)time);
            }
      catch (InterruptedException e) {}
}
```

3. Program the robot to describe a triangular path. Present three different versions of the program using the three types of loop structures: “while”, “do-while” and “for”.

Program with while:

```
int i = 0;
while (i < 3) {
        goForward();
        turnAngle(60);
        i++;
}
stop();
```

Program with do-while:

```
int i = 0;
do{
        goForward();
        turnAngle(60);
        i++;
}while(i<3);
stop();
```

Program with for:

```
for(int i = 0; i < 3; i++){

        goForward();
        turnAngle(60);
}
stop();
```

4. In the examples and exercises done in class, driving the robot is accomplished through time-controlled maneuvers. What are the disadvantages of using this technique (reference to how the effectiveness in driving the robot can be affected by external factors such as the battery or the surface type)?

R: Como as manobras são controladas pelo tempo, quanto mais tempo ele demorar a realizar a manobra mais bateria vai gastar e consequentemente a sua autonomia(tempo que dura sem carregar diminui). A superficie onde ele se desloca também afecta porque quanto mais atrito houver, mais tempo demora a percorrer a distância pretendida.

5. Consider that a new class called MotorI2C that implements the Motor interface was created. Consider that this new class lets you control a certain type of motors through the I2C bus of the IntelliBrain controller.

Consider that the class has the following constructor: MotorI2C (I2CMaster I2c).

What should we change in the program of exercise 1 so that we could use the same program to control a robot equipped with such motors.

Program:

```
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.Motor;
import com.ridgesoft.robotics.MotorI2C;
public class ManeuverWithServo {
    private static Motor leftMotor;
    private static Motor rightMotor;

  public static void main(String args[]) {
            System.out.println("Maneuver I");
            leftMotor = new MotorI2C(IntelliBrain.getI2CMaster());
            rightMotor = new MotorI2C(IntelliBrain.getI2CMaster());
```

```
                leftMotor.setPower(8);
                rightMotor.setPower(8);
                try {
                        Thread.sleep(2500);
                 } catch (InterruptedException e){}
                leftMotor.stop();
                rightMotor.stop();
        }
}
```

Describe the advantages of organizing the classes that control different types of motors by implementing a common interface class like the Motor class.

R: Uma classe interface é uma classe abstrata, ou seja, apenas tem métodos declarados, não faz a implementação dos métodos. A classe que implementa a classe interface fica com a responsabilidade de implementar os metodos declarados.

No caso na classe interface Motor, se pretender adicionar um motor novo, posso fazer de forma simples sem ter que efetuar grandes alterações no código. Esse novo motor só tem que implementar as funções da classe interface Motor.