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

Nº 3			
Group: 3.1 ←Identify your group			
Number	1012164	Name	Carina Tomé
Number	1012208	Name	Dário Ribeiro
Number	1012180	Name	Pedro Sanches
Number	1012396	Name	Rui Manta

Fill in the header record with the names and numbers of the group members and add "(missing)" next to the name, in case any member of the group missed the class. After finishing the laboratorial work, create a PDF doc called "Grupo X.X – LW X.PDF" submit it via the form at http://bit.ly/2TL0lKS.

If the answers to the questions is Java code, you must use color black and font Courier New to write the code. Also, indent all the code. A penalty of 50% will be applied if you forget to do this.

1. Complete the following text about the characteristics of DC motors.
   The nominal voltage is the supply voltage of the motor as recommended by the manufacturer and is measured in volts The motor can operate at voltages lower than the recommended, but with less power The nominal current of a motor is the peak current drawn by the motor and is measured in amperes The current consumption is minimal when the motor runs without load and is maximum when the load is such that it causes the motor to stop. The electronic circuit that controls a motor, must support that maximum current.

2. Complete the following text about servomotors.
   A servomotor is composed by an electronic circuit which controls a motor of the type DC which is mechanically connected to gears to convert the motor speed into more torque A typical servomotor can rotate its axis within a 180 degrees range. The servomotor axis position is controlled by sending to the servomotor a PWM electronic signal. This signal must have a frequency of 50Hz and the amplitude of the pulses must vary between 1ms and 2ms that correspond to the 180 degrees The electronic circuit of the servomotor detects the amplitude of pulses of the received signal and rotates the axis to the correspondent position. To determine in which angular position the shaft is located, the servomotor uses a pontentiometer as a position sensor that is connected to the shaft of the motor. As the shaft rotates, the position sensor also rotates and generates a feedback signal that informs the angular position of the shaft to the electronic circuit.

3. Create a function called turnAngle(int angle), to make the robot rotate the number of degrees passed as a parameter. Consider that the angle can be positive or negative. When positive the robot should rotate counter-clockwise, and when negative the robot should rotate clockwise.
   Tip: Start by find out the time needed for the robot to rotate 180 degrees, at a chosen speed (choose a low speed to avoid inertial forces). Next, use that result to create a formula, based on cross-multiplication, to convert the value of the angle passed as a parameter to the value of time the robot needs to rotate. Finally, implement the code that makes the robot rotate during the time given by the formula.

Program:

```
import com.ridgesoft.intellibrain.IntelliBrain;
import com.ridgesoft.robotics.Servo;

public class IntelliBrainServo {
  public static void main(String args[]) {
        turnAngle(180);
  }
public static void turnAngle(int angle){
        int angulo = angle*(250/180);
        angle = (int) angulo;
        try {
           Servo leftServo = IntelliBrain.getServo(1);
           Servo rightServo = IntelliBrain.getServo(2);
           if(angle > 0) {
             leftServo.setPosition(40);
             rightServo.setPosition(40);
             Thread.sleep(angle);
           } else {
             leftServo.setPosition(60);
             rightServo.setPosition(60);
             Thread.sleep(angle);
           }
      } catch (Throwable t) {
      }
  }
}
```

4. Create a function called goDistance(int distance) to control the robot to move straight forward, approximately the distance passed by the parameter “distance”. Use the same strategy you used for question 3.

   Code:

```
public static void goDistance(int distance) {
        int distancia = distance * (2000/51);
        distancia = (int) distancia;
        leftmotor.setPower(16);
        rightmotor.setPower(16);
        wait(distancia);
  }
```

5. Create a function called move(int power, int offset) that can control the robot to move in a straight line or in a curve, forward or backward. Explain how the values of the parameters relate to the moves.

   Tip: Pass 2 parameters to the function. The first is the base power, and the second is an offset to sum or subtract to the base power (depending on its signal). With just these two parameters we can control the robot to maneuver as pretended.

Function:

```
public static void move(int power, int offset) {
leftMotor.setPower(power+offset);
rightMotor.setPower(power-offset);
}
```

6. Write a function called drawFive(int length) that combines the functions already developed (goForward, rotateAng and move), to make the robot describe, approximately, a path define by the number "5". The parameter "length" is the desired length of the straight lines of the "5". Use comments in the code to identify the parts of the code that correspond to the parts of the path.

Code:

```
public static void turnAngle(int angle){
        double angulo = angle * 180;
        double angled = angulo / 180;
        angle = (int) angled;
        System.out.println(""+angle);
              try {
        Servo leftServo = IntelliBrain.getServo(1);
        Servo rightServo = IntelliBrain.getServo(2);
        if(angle> 0) {
        leftServo.setPosition(40);
        rightServo.setPosition(40);
        Thread.sleep(angle);
        } else {
        leftServo.setPosition(60);
        rightServo.setPosition(60);
        Thread.sleep(angle);
        }
        } catch (Throwable t) {
        }

  }
  public static void goDistance(int distance) {
        int distancia = distance * (2000/51);
        distancia = (int) distancia;
        leftmotor.setPower(16);
        rightmotor.setPower(16);
        wait(distancia);
  }
```

```
public static void move(int power, int offset) {
      leftmotor.setPower(power - offset);
      rightmotor.setPower(power + offset);
}

public static void stop() {
      leftmotor.stop();
      rightmotor.stop();

}


public static void drawFive(int length) {

      goDistance(length);
      turnAngle(90);
      wait(250);
      goDistance(length);
      turnAngle(90);
      wait(250);
      move(10,-8);
      wait(2000);
      stop();

}
```

7. 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? (Think about how the effectiveness in driving the robot can be affected by external factors such as the battery or the surface type)?
   R: This technique has many problems, especially when we change the ground. It's not a viable option unless we know exactly where it will perform. Additionally, the only know to calculate some things is by trial and error.

8. Consider that a new class called MotorI2C was created that implements the Motor interface. 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 ManeuverWithMotor so that we could use the same program to control a robot equipped with such motors. Present only the changes we need to make.
   Code:

```
public static void main(String[] args) {
leftMotor = new MotorI2C(IntelliBrain.getI2CMaster());
rightMotor = new MotorI2C(IntelliBrain.getI2CMaster());
```

...