**Quiz 02 – Practice Questions**

1. Classify the following robot control programs as “open loop” or “closed loop”.
   1. A DVD player that uses a proportional controller to ensure that the disk spins at a fixed rate of revolutions per minute.
   2. A differential drive robot that drives straight by spinning both wheels at the same speed in the same direction.
   3. A differential drive robot that uses a gyro (or a magnetic) sensor to check whether it drives straight and takes corrective actions if necessary.
   4. An ICBM (Intercontinental Ballistic Missile) (look this up if necessary)
   5. A cruise missile (look this up if necessary)
2. Are the following sensors “active” or “passive” sensors
   1. compass
   2. Radar
   3. GPS
   4. The camera in your smart phone (during daylight)
   5. The camera in your smart phone at night (using its flash)
   6. Seismograph
   7. Telescope
3. A submarine can use active sonar as well as passive listening sensors to locate other submarines. Which method can better detect opposing boats? What is a big disadvantage of using this better method?
4. We discussed “compile-time errors”, “runtime errors”, and “logical errors”. At which point of creating and executing a robot program does each error occur? Why is a runtime error much harder to detect and fix than a compile-time error? What about a logical error, is it harder or easier to fix than a compile-time error?
5. There are three variations of the conditional if statement. What is the syntax of each?
   1. Simple if
   2. if with alternative
   3. multiple if-else statement
6. Which variation of the if statement applies to the following situation. Sketch the code, using English phrases like “goStraight()” to indicate some action if necessary:
   1. A robot detects a colored spot. If it is red, it should turn left, if it is green, turn right, if yellow, go straight
   2. If a robot detects an obstacle ahead, it should turn by 90 degrees
   3. A robot turns a control light green if it still has enough battery power, otherwise it turns light to red
7. What construction do you use to repeatedly execute code without having to re-type it?
8. What is an “infinite loop” and how do you stop it should one execute on your brick?
9. What is an “unnamed constant”, why should you not use them, and with what should you replace them.
10. How do you define a constant and where do you put that definition?
11. Why is it better to use named constants instead of plain numbers in a program?
12. What is he output of this code sequence:

int count = 0;  
while (count < 5)  
{  
 System.out.println(count);  
 count = count + 2;  
}

1. What is the output of this code sequence assuming that the variable test has value of 85 when the code executes:

if (test <= 100)

{

System.out.println(“A”);  
}  
else if (test < 90)  
{

System.out.println(“B”);

}

else if (test < 80)  
{

System.out.println(“C”);

}

else

{

System.out.println(“worse than C (sorry)”);

}

1. Consider the multiple if statement shown in number 10. Its intention is to write the letter grade corresponding to the current value of test. However, that statement is logically wrong (see #10). Fix it, i.e. create multiple if statements that will print out the proper letter grade for the current value of test.
2. Consider the following code. While it is correct and will function correctly, it is still considered poor programming. Why? Rewrite the code to fix the problem:

if (distance < 0.15)  
{  
 slowDown();  
}  
else

{  
 speedUp();  
}

1. Create some code that
   1. Prints the numbers 0 to 9
   2. Prints the numbers 1 to 10
   3. Prints the even numbers less than 11
2. Suppose your robot has a touch sensor, facing forward, and a distance sensor facing right. Place the robot parallel to a wall with the wall on the right of the robot. Create the main method of a program that makes the robot follow the wall. Assume the following methods are available and work as advertise:

*// Returns true if robot is getting too close to the wall, false otherwise*

public static boolean isTooCloseToWall()

*// Returns true if robot is getting too far away from the wall, false otherwise*

public static booean isTooFarFromWall()

*// Returns true if the path straight ahead is blocked by some obstacle*

public static boolean pathBlocked()

*// Moves robot forward while turning away from the wall*

public static void turnAwayFromWall()

*// Moves robot forward while turning towards from the wall*

public static void turnTowardsWall()

*// Turns robot 90 degrees to the left*

public static void turnLeft()

*// Moves robot forward*

public static void goForward()

Hint: use one of the variants of the conditional if statement inside a loop

1. What is a “proportional controller” in your own words? Is it open or closed loop?