AI Driver

The challenges!

Contents

Download the project files here: https://github.com/TempeHS/AIDriver_Challanges

Lucid Chart Arduino Flow Chart: https://lucid.app/lucidchart/b4213029-4756-433a-a3a0-89985c82757d/edit?invitationId=inv_0fb9e1d6-98cc-4ee9-badd-92e63c0ed0d6

Challenges Video Introduction: https://www.youtube.com/watch?v=6tkc6a3uOuw&list=PL59nymGkZ9KuRvR-xONV86-UJgiKemQM8&index=10

Challenge 1: Drive a perfect circle in a loop

Your challenge is to get your robot to drive in a perfect circle continuously less than 1.5m in diameter.

Criteria for success:

1. My robot drives a perfect circle.
2. My robot repeats the same circle.

Things to think about:

1. This challenge is about control and balance, the fastest is not the best option.

Setup & Loop methods explained: https://www.youtube.com/watch?v=d1f5BCO2DDY

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:

In the loop method add

```
mrJonesDriving->driveForward(125,125);
```

Step 2:

Adjust the speed values of the left and right wheels between 0 (stop) and 255 (fastest) to get your robot to drive in a perfect circle.

Step 3:

Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria

Challenge 1 extension, I can...

- [ ] Drive a circle both directions.
- [ ] Drive a circle in reverse.
- [ ] Drive different sized circles.
- [ ] Stop and pause at each loop.
- [ ] Drive a figure 8.

Challenge 2: Drive 2m in a straight line and stop

Your challenge is to get your robot to drive exactly 2m in a straight and stop.

Criteria for success:

1. My robot drives in a straight line.
2. My robot comes to a standstill at 2m.

Things to think about:

1. Your car does not have precision motors, you will need to adjust the speed and duration to get them running perfectly straight.
2. Even though the code runs in a loop you need to add a logic control structure, so it only runs once then is skipped by the loop.

Let's start by trying the algorithm: https://blockly.games/maze?lang=en&level=1&&skin=0

If control logic explained: https://youtu.be/SKQzljqd3cQ?t=185

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:

Above the setup method add (these are called global variables)

```
bool myRepeat = true;
```

In the loop method add

```
if (myRepeat) {

  // Move forward
  myRepeat = false;
} else {
  // stop
}
```

Step 2:

Look in the notes.ino file for examples of different methods you can use to control your robot to overcome the challenge.

Step 3:

Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria.

Challenge 2 extension, I can...

- [ ] Progressively accelerate my robot to full speed.
- [ ] Progressively deaccelerate my robot to a stop.

Challenge 3: Drive a perfect 1m square and stop

Your challenge is to get your robot to drive 1m, turn 90 degrees repeated 4 times.

Criteria for success:

1. My robot drives in a straight line for 1m
2. My robot comes to a standstill then turns 90 degrees
3. My robot does this exactly 4 times

Things to think about:

1. Your car does not have precision motors, you will need to adjust the speed and duration to get them running perfectly straight.
2. Even though the code runs in a loop you need to add a logic control structure so it only runs once then is skipped by the loop.
3. What would happen if you din't increment the counter?

Let's start by trying the algorithm: https://blockly.games/maze?lang=en&level=6&&skin=0
While loop explained: https://www.youtube.com/watch?v=j0XMHydEOdM

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:
Above the setup method add (these are called global variables)

```
int myCount = 0;
```

In the loop method add

```
while (myCount < 4) {

  // Move forward
  myCount = myCount + 1;
}
  // stop
```

Step 2:
Look in the notes.ino file for examples of different methods you can use to control your robot to overcome the challenge.

Step 3:
Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria

Challenge 3 extension, I can...

- [ ] I can drive a triangle.
- [ ] I can put the turn and forward commands in sperate functions to call on demand.
- [ ] I can use a for loop to do the exact some challenge.

Challenge 4: Drive 1m, turn 180 degree's and return

Your challenge is to get your robot to drive exactly 1m turn 180 degrees and return to the exact starting point using functions you have written.

Criteria for success:

1. My robot drives in a straight line.
2. My robot comes to a standstill at 2m.

Things to think about:

4. Your car does not have precision motors, you will need to adjust the speed and duration to get them running perfectly straight.
5. Even though the code runs in a loop you need to add a logic control structure so it only runs once then is skipped by the loop.

Let's start by trying the algorithm: https://blockly.games/maze?lang=en&level=6&&skin=0
Functions explained: https://www.youtube.com/watch?v=Wx6-TPtTavI

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:

In the setup method add

```
driveOneMeter(); //this will call the function
```

Below the loop method (make sure outside the last '}':

```
void driveOneMeter () {
 // your code here
}
```

Step 2:

Look in the notes.ino file for examples of different methods you can use to control your robot to overcome the challenge.

Step 3:

Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria.

Challenge 4 extension, I can...

- [ ] Move the function call from the setup to the loop, first predict then test
- [ ] Then see how few lines of code you can write to make it go up and back constantly.
- [ ] Progressively accelerate and/or deaccelerate my robot to full speed.

Challenge 5: Collision Prevention

Your challenge is to write an algorithm so your robot comes to a stop when it is about to collide, when no collision is detected, it will drive forward.

Criteria for success:

1. My robot drives continuously in a straight line.
2. When an object is placed in front of my robot it safely comes to a complete stop.

Things to think about:

1. How can you use functions, and the control structures you have learnt about to perform this task efficiently?
2. Note: the Ultrasonic Sensor returns '0' if no objects for 3m so make sure your testing arena is not too large.

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:

In the loop method add

```
if (distanceRanger.read() >= 100) {
  // do this
} else {
  // do this
}
```

Step 2:

Look in the notes.ino file for examples of different methods you can use to control your robot to overcome the challenge.

Step 3:

Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria.

Challenge 5 extension, I can...

- [ ] Add options for different reactions if an object is very close or just near.
- [ ] Progressively accelerate and or decelerate my robot to full speed.

Challenge 6: Drive, stop and then reverse to the starting location

Your challenge is to write an algorithm that uses the ultrasonic sensor to drive towards an object stop 50mm away then reverse back to the starting location.

Criteria for success:

1. I can pick a random distance between 05 and 3m from an object.
2. My robot will drive towards the object and stop 50mm away.
3. My robot will reverse to the starting point.

Things to think about:

1. How can you use functions, and the control structures you have learnt about to perform this task efficiently?
2. Think about what data you will need to store and use in your algorithm.
3. Note: the Ultrasonic Sensor returns '0' if no objects for 3m so make sure your testing arena is not too large.

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:

Create the global variables you will need above the setup() method.

Step 2:

Apply the skills you have learnt so far and look in the notes.ino file for examples of different methods you can use to control your robot to overcome the challenge.

Step 3:

Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria.

Challenge 7: Maze navigation

Your challenge is to apply all that you have learnt to write an algorithm so your robot can explore a maze without colliding with walls. It should be able to identify and respond to a dead end and test different options in a multiway junction before navigating them.

Criteria for success:

1. My robot drives in a straight line.
2. My robot comes to a standstill at 2m.

Things to think about:

1. This is an extremely challenging task, you should start by getting one thing working at a time and build you application progressively testing at each stage.
2. If you can try and abstract set tasks into functions then call them as you need from the loop() method.

Let's start by trying the algorithm: https://blockly.games/maze?lang=en&level=10&&skin=0

TURN BATTERY PACK OFF BEFORE CONNECTING USB TO COMPUTER

Step 1:

Look in the notes.ino file for examples of different methods you can use to control your robot to overcome the challenge.

Look at all you other algorithms and work out what you can repurpose.

There are many ways to solve this problem.

Step 3:

Once your algorithm is working model it in lucid charts before adding it to your folio with an evaluation of the success criteria.