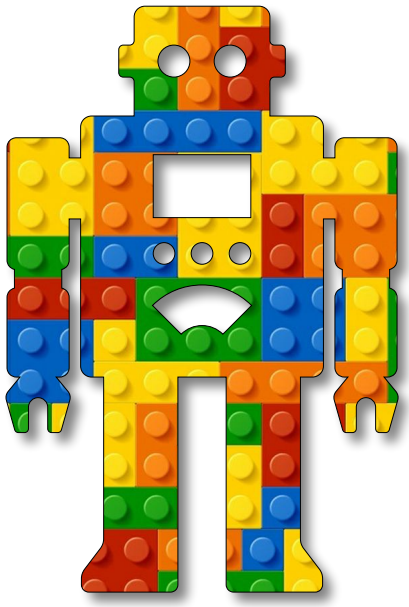
Intelligent Machines Module A Unit #5 random & arrays



Content

Module A Unit #5 random & arrays

Introduction to random and arrays

Arrays: Arrays are a very powerful tool for collecting, storing, and manipulating data. This is a simple introduction to its use and purpose.

Random: The `random()` function gives you a random number. Although random for most purposes, it does have a pattern to it. There are ways to improve the randomness using `randomSeed()`. It can take a random value for one of the pins (noise), so that each time the sketch is run, it chooses a different `randomSeed()`.

Sketch A5.1 a fast blink

We will start with a basic sketch that you are familiar with. This will blink reasonably fast.

Arduino sketch

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  blink(250);
}

void blink(int delayPeriod)
{
  digitalWrite(13, HIGH);
  delay(delayPeriod);
  digitalWrite(13, LOW);
  delay(delayPeriod);
}
```

Notes

Rapid blinking!

Sketch A5.2 hard code

We will hard-code the `delayPeriod` for when the LED is off (`LOW`). Now it is on for `250` milliseconds and off for `100` milliseconds.

Arduino sketch

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  blink(250);
}

void blink(int delayPeriod)
{
  digitalWrite(13, HIGH);
  delay(delayPeriod);
  digitalWrite(13, LOW);
  delay(100);
}
```

Challenge

Try a variety of combinations.

Sketch A5.3 adding an array

We are going to add an array for how long the LED is on for. The name of the array is `durations[]`. The data is stored in curly brackets `{}`, separated by commas. We are going to have the LED on for `eight` different periods of time (`elements` in the array).

Arduino sketch

```
int durations[] = {100, 100, 2000, 2000, 100, 100, 2000, 2000};


void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  blink(250);
}

void blink(int delayPeriod)
{
  digitalWrite(13, HIGH);
  delay(delayPeriod);
  digitalWrite(13, LOW);
  delay(100);
}
```

🗒️ Notes

Nothing is going to change yet.

🌻 Challenge

You can create your own array of elements.

🛠️ Code Explanation

`durations[]`	An empty array called durations. The [] brackets denote an array, this is then filled with the elements in the {} braces.

Sketch A5.4 calling the elements

Now we need to call each element of the array in turn. The counting in arrays is different to the way we count. The counting starts with 0, then 1, 2, 3, 4, 5… then two rather than start with 1, 2, 3, 4, 5… so the first element in the array is position or index 0 (not 1).

Now we need to loop through each one in turn and we can do this with a `for()` loop. Each `i` is an `index`. So `index[0]` is `100`, `index[1]` is `100`, `index[2]` is `2000`, `index[3]` is `2000`, `index[4]` is `100` and so on.

This translates to `durations[0]` is `100`, `durations[1]` is `100`, `durations[2]` is `2000` and so on, looping through one at a time till it gets to the last one and starts all over again.

Arduino sketch

```
int durations[] = {100, 100, 2000, 2000, 100, 100, 2000, 2000};


void setup()
{
  pinMode(13, OUTPUT);
}


void loop()
{
  for (int i = 0; i < 8; i++)
  {
    blink(durations[i]);
  }
}


void blink(int delayPeriod)
{
  digitalWrite(13, HIGH);
  delay(delayPeriod);
  digitalWrite(13, LOW);
  delay(250);
}
```

🗒️ Notes

Each value is passed onto the `delayPeriod` for the `blink()` function. What you should see is two short blinks followed by two longer blinks, and then two short blinks and then two long blinks.

🌻 Challenge

If you aren't sure exactly how big the array is but need it for the `for()` loop, you cannot use `durations.length` as you could with p5.js; instead, use the `sizeof()` function (I am not going to go into why it is this way!!). Try it:

```
for (int i = 0; i < sizeof(durations)/sizeof(durations[0]); i++)
{
  .....
}
```

Sketch A5.5 a very basic sketch

❗ Starting with a very basic new blink sketch.

Arduino sketch

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

🗒️ Notes

If you are not convinced that the code has produced the changes you have entered, then press the RESET button to start the programme from the beginning.

Sketch A5.6 a random variable

Introduce a new variable called `delayRandom` and give it an initial value of `1000`. It is always a good idea to give a variable an initial value.

Arduino sketch

```
int delayRandom = 1000;

void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(delayRandom);
  digitalWrite(13, LOW);
  delay(delayRandom);
}
```

Notes

This should provide no change whatsoever over the previous sketch.

Sketch A5.7 a random delay

Now we change the 1000 milliseconds to a random number between 0 and 1000. It will change every time it loops through.

Arduino sketch

```
int delayRandom = 1000;

void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  delayRandom = random(1000);
  digitalWrite(13, HIGH);
  delay(delayRandom);
  digitalWrite(13, LOW);
  delay(delayRandom);
}
```

🗒️ Notes

It will blink at random rates, as if in Morse code.

🌻 Challenge

Create another random variable for the period the LED is off (LOW). It will create a very random effect.

🛠️ Code Explanation

random(1000);	Gives you a random number between 0 and 1000

Sketch A5.8 random limits

We can create upper and lower limits, so now it has a random number between 500 and 2000 milliseconds.

Arduino sketch

```
int delayRandom = 1000;

void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  delayRandom = random(500, 2000);
  digitalWrite(13, HIGH);
  delay(delayRandom);
  digitalWrite(13, LOW);
  delay(delayRandom);
}
```

🗒️ Notes

Does just what it says on the tin.

🛠️ Code Explanation

`random(500, 2000);`	Returns a random number between 500 and 2000

Sketch A5.9 basic delay

❗ Start with a new basic sketch with the delay set to `1000`.

Arduino sketch

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

Notes

Feels like we have been here before.

Sketch A5.10 ten element array

We add an array of 10 elements. This is a blank or empty array, effectively 10 zeros.

Arduino sketch

```
int durations[10];


void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

🗒️ Notes

It is considered good form to give the array an initial size.

🛠️ Code Explanation

`int durations[10];`	This array is effectively empty but it has been given the dimensions of 10 elements.

Sketch A5.11 filling the array

In `setup()`, we fill each element with a random number between `0` and `1000`. We do this by looping through ten times, each time selecting a random number.

Arduino sketch

```
int durations[10];

void setup()
{
  pinMode(13, OUTPUT);
  for (int i = 0; i < 10; i++)
  {
    durations[i] = random(1000);
  }
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

🗒️ Notes

This puts `10` random numbers into the array (just once) in `setup()`.

🛠️ Code Explanation

`durations[i] = random(1000);`	As it cycles through the loop, [i] starts at zero and increments by 1 each loop. So for each position in the array it puts a random number from 0 to 1000 into that array. The [i] value is the index reference for the array, counting starts at zero not 1.

Sketch A5.12 loop in a function

Now we need to read from each element in a similar loop inside the `void loop()` function.

Arduino sketch

```
int durations[10];


void setup()
{
  pinMode(13, OUTPUT);
  for (int i = 0; i < 10; i++)
  {
    durations[i] = random(1000);
  }
}


void loop()
{
  for (int i = 0; i < 10; i++)
  {
    digitalWrite(13, HIGH);
    delay(durations[i]);
    digitalWrite(13, LOW);
    delay(durations[i]);
  }
}
```

🗒️ Notes

You should see the same pattern of blinks every `10` seconds.

🌻 Challenge

Create another array for when the LED is `off`.

🛠️ Code Explanation

`delay(durations[i]);`	This is the reverse of the previous operation. It goes through the array one element at a time starting at index 0 through to index 9, it reads its value and that is the length of the delay.

Sketch A5.13 an array of floats

A new sketch illustrates how to create an array of random `floats` to two decimal places by default.

Arduino sketch

```
float myArray[10];
bool repeat = true;

void setup()
{
  Serial.begin(9600);
  for (int i = 0; i < 10; i++)
  {
    myArray[i] = float(random(100))/100;
  }
}

void loop()
{
  delay(3000);
  if (repeat == true)
  {
    for (int i = 0; i < 10; i++)
    {
      Serial.print(i);
      Serial.print(" ");
      Serial.println(myArray[i]);
    }
  }
  repeat = false;
}
```

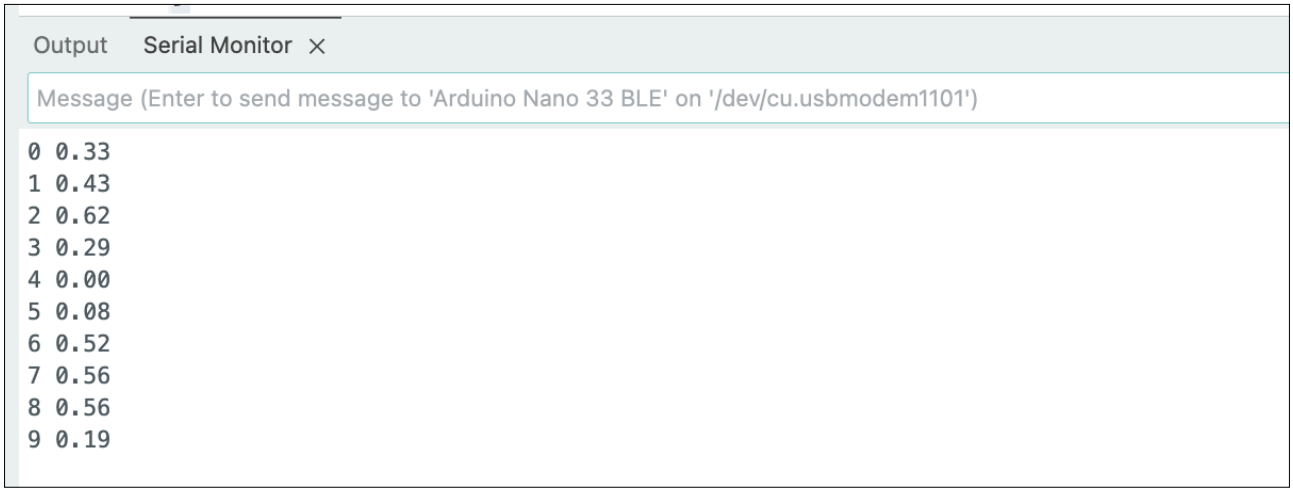
Notes

The delay of 3 seconds is necessary for the serial monitor to catch up, otherwise it misses the early part of the array.

🛠️ Code Explanation

`float myArray[10];`	Create an array of 10 elements
`bool repeat = true;`	We need a mechanism to only send data once to the serial monitor, initialise boolean to true
`myArray[i] = float(random(100))/100;`	This creates a random number between 0-100 and then divides that number to give us a number to two decimal places
`delay(3000);`	Gives time for the serial monitor to catch up
`if (repeat == true)`	Only goes through the conditional loop if true
`Serial.print(i);`	Prints the index value starting at zero
`Serial.print(" ");`	Leave a space
`Serial.println(myArray[i]);`	Print the value in the array at that index, then carriage return (new line)
`repeat = false;`	Negates the condition

Figure A5.13



Sketch A5.14 to five decimal places

We simply increase the random number generated and divide it by that amount.

Arduino sketch

```
float myArray[10];
bool repeat = true;

void setup()
{
  Serial.begin(9600);
  for (int i = 0; i < 10; i++)
  {
    myArray[i] = float(random(100000))/100000;
  }
}

void loop()
{
  delay(3000);
  if (repeat == true)
  {
    for (int i = 0; i < 10; i++)
    {
      Serial.print(i);
      Serial.print(" ");
      Serial.println(myArray[i], 5);
    }
  }
  repeat = false;
}
```

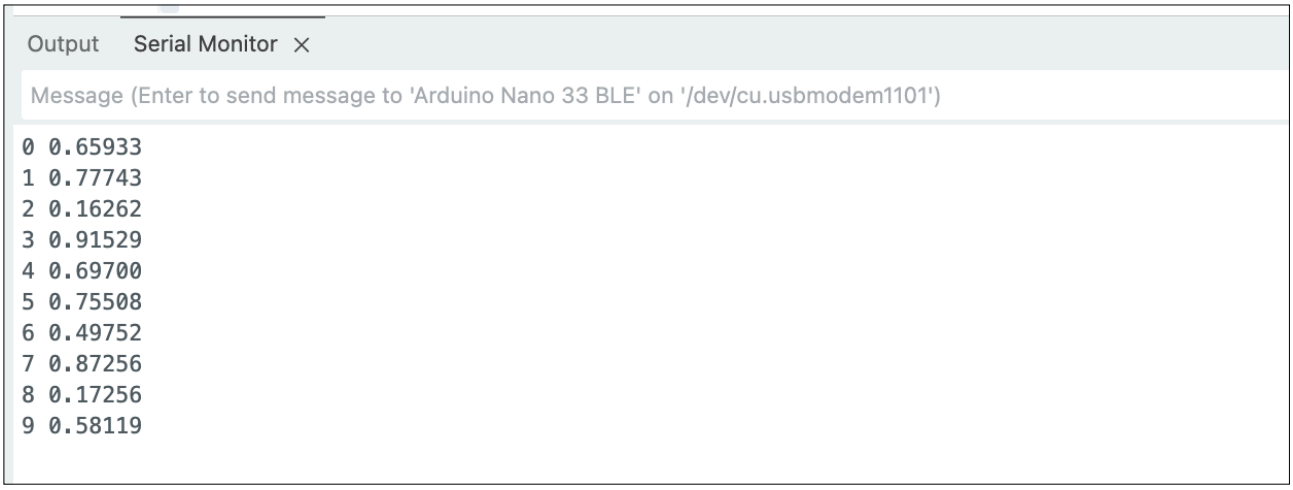
Notes

We do need to tell it we want 5 decimal places. If you keep resetting the board, you will notice that you get the same random numbers each time. There is a way round this.

🛠️ Code Explanation

`myArray[i] = float(random(100000))/100000;`	Random to 100,000 and divide by 100,000
`Serial.println(myArray[i], 5);`	To five decimal places

Figure A5.14



Sketch A5.15 randomising the random

Because it will pick the same random values each time, we can change the random values by calling a particular `randomSeed()` such as `randomSeed(3)` for instance. This will give you the same values each time but will be different for other `randomSeeds(x)`. One way to randomise the `randomSeed()` is to gate the seed value for one of the pins which is fluctuating slightly all the time (noise in the system), a bit like static on a TV (if you can remember).

Arduino sketch

```
float myArray[10];
bool repeat = true;

void setup()
{
  Serial.begin(9600);
  randomSeed(analogRead(4));
  for (int i = 0; i < 10; i++)
  {
    myArray[i] = float(random(100000))/100000;
  }
}

void loop()
{
  delay(3000);
  if (repeat == true)
  {
    for (int i = 0; i < 10; i++)
    {
      Serial.print(i);
      Serial.print(" ");
      Serial.println(myArray[i], 5);
    }
  }
  repeat = false;
}
```

🗒️ Notes

Every time you reset the board, you will get a different set of random numbers.

🛠️ Code Explanation

`randomSeed(analogRead(4));`	Chose a pin that isn't being used for anything

Figure A5.15a

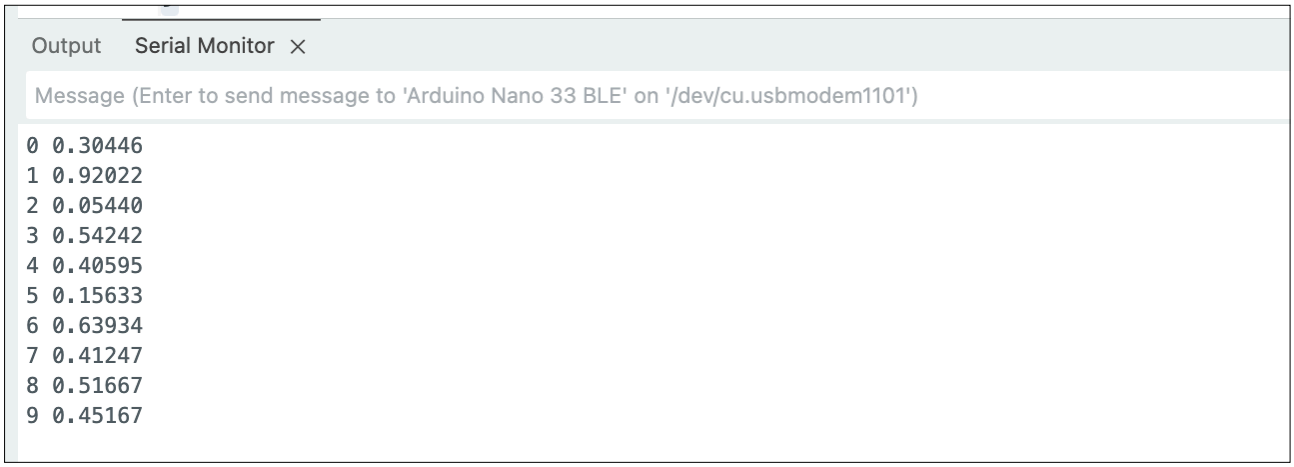


Figure A5.15b