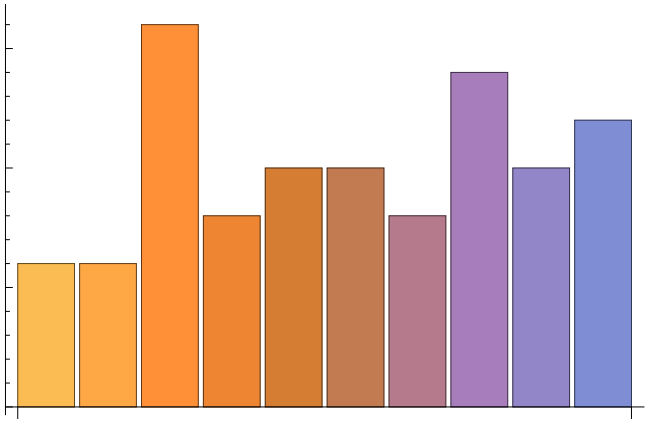
```
(* Adam Beck *)
(* Simulate equipartition for the number e at levels 1 and 2 *)

(* Decimal expansion of e, small, level 1 *)
e = RealDigits[E, 10, 50]; (* number, base, length of digits *)
smallE = e[[1]]; (* I now have my small amount of digits for e *)

(* I am writing this function to compute the sum of a given digit "n" in e *)
f[n_] := N[Sum[If[{smallE[[k]]} == {n}, 1, 0], {k, 1, Length[smallE]}] / (Length[smallE])]

(* An empty list will store the equipartition ratios *)
equipartition = {};

(* Loop from 0 to 9, and find the equipartition ratio
 for each number. Append this to the equipartition list *)
For[i = 0, i < 10, i++, AppendTo[equipartition, f[i]];];
equipartition
```

```
{0.06, 0.06, 0.16, 0.08, 0.1, 0.1, 0.08, 0.14, 0.1, 0.12}
```

```
(* Graph of the equipartitions *)
BarChart[{equipartition}]
```

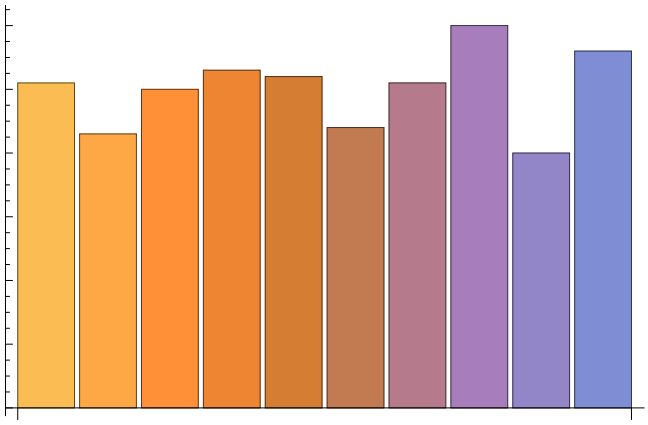


```
(* We can see how the equipartitions are not similar. This is expected,
as I only chose 50 digits of e *)

(* Decimal expansion of e, medium, level 1 *)
e = RealDigits[E, 10, 500]; (* Obtain a medium amount of digits for e *)
mediumE = e[[1]];

(* I am writing this function to compute the sum of a given digit "n" in e *)
f[n_] := N[∑_{k=1}^{Length[mediumE]} If[{mediumE[[k]]} == {n}, 1, 0] / (Length[mediumE])]

equipartionMedium = {} (* An empty list will store the equipartition ratios *)

(* Loop from 0 to 9, and find the equipartition ratio
 for each number. Append this to the equipartition list *)
For[i = 0, i < 10, i++, AppendTo[equipartionMedium, f[i]];];
equipartionMedium
```

```
{}
```

```
{0.102, 0.086, 0.1, 0.106, 0.104, 0.088, 0.102, 0.12, 0.08, 0.112}
```

```
(* Graph of the equipartitions *)
BarChart[{equipartionMedium}]
```



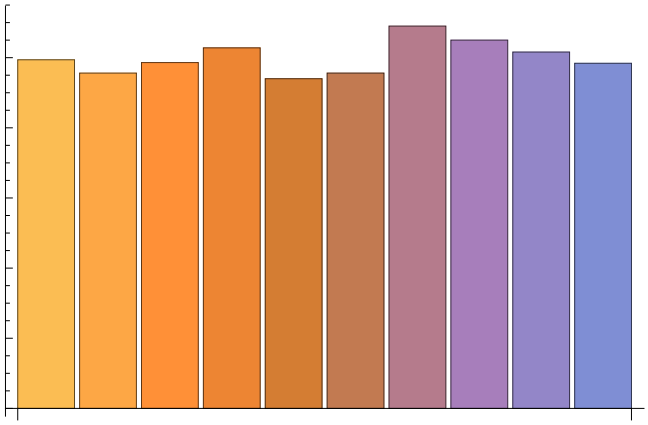
```
(* As one can see, the graph is starting to equal out. This
 is expected as we took a medium lengthed list of digits *)

(* Decimal expansion of e, large, level 1*)
e  = RealDigits[E, 10, 5000]; (* compute a large amount of digits of e *)
largeE = e[[1]];

(* I am writing this function to compute the sum of a given digit "n" in e *)
f[n_] := N[∑_{k=1}^{Length[largeE]} If[{largeE[[k]]} == {n}, 1, 0] / (Length[largeE])]
```

```
(* Find the equipartition ratios of the digits 0-
 9 and store them in the equipartition list *)
equipartitionLarge = {}
For[i = 0, i < 10, i++, AppendTo[equipartitionLarge, f[i]];];
equipartitionLarge
```

```
{}
```

```
{0.0994, 0.0956, 0.0986, 0.1028, 0.094, 0.0956, 0.109, 0.105, 0.1016, 0.0984}
```

```
(* Graph the equipartitions *)
BarChart[{equipartitionLarge}]
```

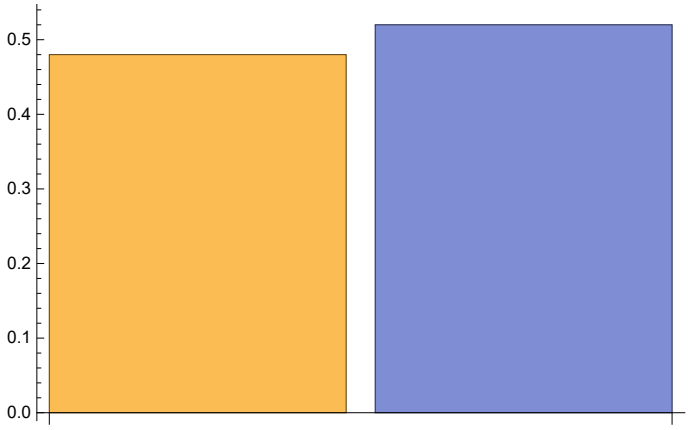


```
(* As one can see,
the graph is far more equalized than the small and medium graph. This is expected, as we
 have a large amount of digits,
and the law of large numbers tells us that this outcome was expected. It is evening out
 at about 0.1 for each digit which is expected. *)
```

```
(* Binary expansion of the number e, small, level 1 *)
e = RealDigits[E, 2, 50];
(* Note that the second parameter now computes E in base 2 *)
smallE = e[[1]];

(* Create a list to store the equipartition ratios *)
equipartitionSmall = {}

(* I am writing this function to compute the sum of a given digit "n" in e *)
f[n_] := N[ ∑_{k=1}^{Length[smallE]} If[{smallE[[k]]} == {n}, 1, 0] / (Length[smallE]) ]
(* Loop from 0-
 1 and find the equipartition ratios of those numbers. Append the ratios to the list *)
For[i = 0, i < 2, i++, AppendTo[equipartitionSmall, f[i]];];

equipartitionSmall
```

```
{0.48, 0.52}
```

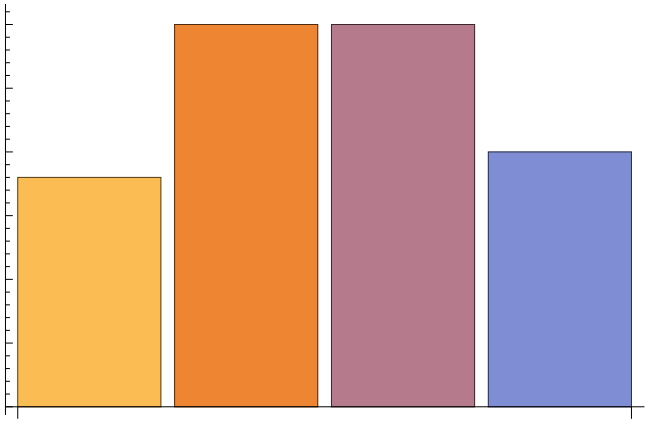
```
(* Graph the equipartitions *)
BarChart[{equipartitionSmall}]
```




```
(* One can note that the equipartitions are close to .50,
which is expected. Even though this is our "small" trial,
the values are fairly equal. We expect
 this to center around .50 for the next two trials *)

(* Binary expansion of the number e, small, level 2 *)

e = RealDigits[E, 2, 50]

smallE = e[[1]]; (* Gets the binary digits for e *)
equipartitionSmall = {}; (* create a list to store the equipartitions in *)

(* This function computes the equipartition
  ratitos for a given two digits "n" and "m". Note
  how we are looping to Length[smallE]-1 and dividing by this amount,
and not the full Length[smallE] *)
f[n_, m_] :=
  N[ Sum[If[{smallE[[k]], smallE[[k + 1]]} == {n, m}, 1, 0], {k, 1, Length[smallE] - 1}] / (Length[smallE] - 1)];
(* Append the equipartitions of all possible two-digit pairs in e *)
AppendTo[equipartitionSmall, f[0, 0]];
AppendTo[equipartitionSmall, f[0, 1]];
AppendTo[equipartitionSmall, f[1, 0]];
AppendTo[equipartitionSmall, f[1, 1]];
equipartitionSmall
```

```
{0.18, 0.3, 0.3, 0.2}
```

```
(* Graph the equipartitions *)
BarChart[{equipartitionSmall}]
```

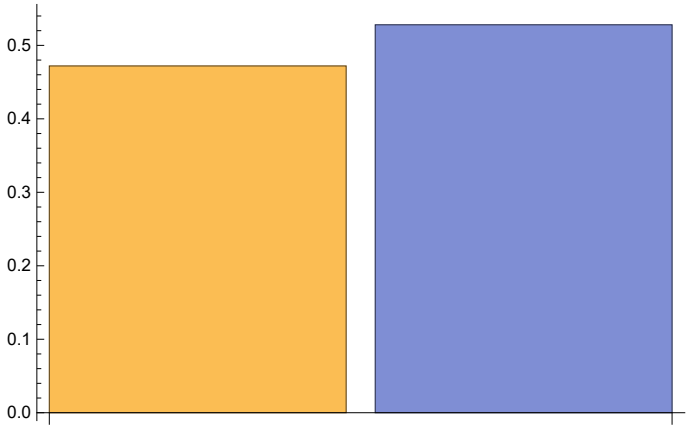


```
(* Unlike at level 1,
this graph is not equalized. It seems random and we would expect it to equalize as
 the trial size increases *)

(* Binary expansion of the number e, medium, level 1 *)
e = RealDigits[E, 2, 500];
smallE = e[[1]]; (* should have been named mediumE *)
equipartitionSmall = {}

(* A function to compute the equipartitions of a given digit "n" in e *)
f[n_] := N[ Sum[If[{smallE[[k]]} == {n}, 1, 0], {k, 1, Length[smallE]}] / (Length[smallE])]

(* Loop from 0–
 1 and find the equipartitions for these values in e and append it to the list *)
For[i = 0, i < 2, i++, AppendTo[equipartitionSmall, f[i]];];
equipartitionSmall
```

```
{}
```

```
{0.472, 0.528}
```

```
(* Graph the equipartitions *)
BarChart[{equipartitionSmall}]
```




```
(* This graph is actually similar to the small trial. We
 expected it to normalize around .50 but it stayed closer to our
 results from the small trial. The law of large numbers tells
 us that the large trial will normalize around .50 *)

(* Binary expansion of the number e, medium, level 2 *)
e = RealDigits[E, 2, 500];
mediumE = e[[1]];
equipartitionMedium = {}
```

```
{}
```

```
(* This function computes the equipartition
  ratitos for a given two digits "n" and "m". Note
  how we are looping to Length[mediumE]-1 and dividing by this amount,
and not the full Length[mediumE] *)

f[n_, m_] := N[
    Sum[If[{mediumE[[k]], mediumE[[k + 1]]} == {n, m}, 1, 0], {k, 1, Length[mediumE] - 1}] / (Length[mediumE] - 1)];

equipartionMedium = {}
```

```
{}
```

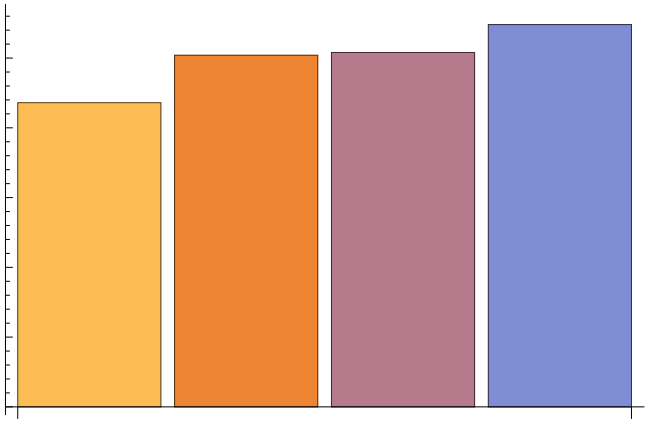
```
(* Append the equipartitions of all possible two-digit pairs in e *)

AppendTo[equipartitionMedium, f[0, 0]];
AppendTo[equipartitionMedium, f[0, 1]];
AppendTo[equipartitionMedium, f[1, 0]];
AppendTo[equipartitionMedium, f[1, 1]];
equipartitionMedium
```

```
{0.218, 0.252, 0.254, 0.274}
```

```
equipartitionMedium
```

```
{0.218, 0.252, 0.254, 0.274}
```

```
(* Graph the equipartitions *)
BarChart[{equipartitionMedium}]
```



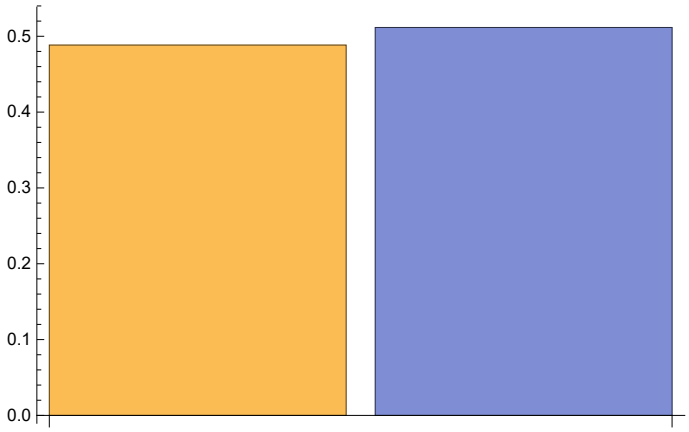
```
(* This graph is more normalized around .25 compared to
 our small trial. This was expected, and we expect this to continue
 for the large trial *)

(* Binary expansion of the number e, large, level 1 *)
equipartitionLarge = {}
```

```
{}
```

```
largeEDigits = RealDigits[E, 2, 5000];

largeE = largeEDigits[[1]]; (* Get the binary digits for e *)

(* A function to compute the equipartitions of a given digit "n" in e *)
f[n_] := N[ ∑_{k=1}^{Length[largeE]} If[{largeE[[k]]} == {n}, 1, 0] / (Length[largeE])];

(* Loop from 0-1 and find the equipartitions
  for these digits in e. Append these ratios to the list *)
For[i = 0, i < 2, i++, AppendTo[equipartitionLarge, f[i]];];

equipartitionLarge
```

```
{0.4884, 0.5116}
```

```
(* Graph the equipartitions *)
BarChart[{equipartitionLarge}]
```



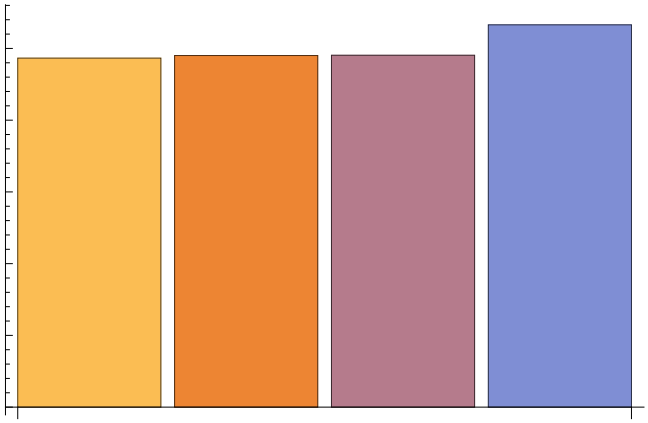

```
(* We can see that these equipartitions are closer
 to .5 compared to the medium trial as predicted by the law
 of large numbers. This was expected *)

(* Binary expansion for the number E, large, level 2 *)
equipartitionLarge = {}
```

```
{}
```

```
largeE = RealDigits[E, 2, 5000];

e = largeE[[1]];

(* This function computes the equipartition
  ratitos for a given two digits "n" and "m". Note
  how we are looping to Length[e]-1 and dividing by this amount,
and not the full Length[e] *)
f[n_, m_] := N[ ∑_{k=1}^{Length[e]-1} If[{e[[k]], e[[k + 1]]} == {n, m}, 1, 0] / (Length[e] - 1)];
(* Append the equipartitions of all possible two-digit pairs in e *)
AppendTo[equipartitionLarge, f[0, 0]];

AppendTo[equipartitionLarge, f[0, 1]];

AppendTo[equipartitionLarge, f[1, 0]];
AppendTo[equipartitionLarge, f[1, 1]];

equipartitionLarge
```

```
{0.243249, 0.245049, 0.245249, 0.266453}
```

```
BarChart[{equipartitionLarge}]
```



```
(* This trial was equalized more around .25 than the medium lengthed trial,
which was predicted from the
 law of large numbers. While still not perfect,
it serves as a good indicator that e, in binary, has a perfect ratito of
 two digit pairs *)

(* Decimal expansion for the number e, small, level 2 *)

e = RealDigits[E, 10, 50];

smallE = e[[1]];

equipartitionSmall = {}
```

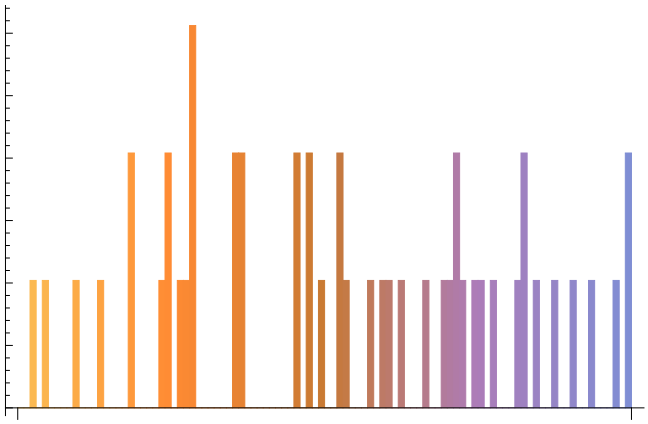
```
{}
```

```
(* This function computes the equipartition
  ratitos for a given two digits "n" and "m". Note
  how we are looping to Length[smallE]-1 and dividing by this amount,
and not the full Length[smallE] *)

f[n_, m_] :=
  N[ Sum[If[{smallE[[k]], smallE[[k + 1]]} == {n, m}, 1, 0], {k, 1, Length[smallE] - 1}] / (Length[smallE] - 1)];


(* Loops from 00 to 99 which are all
 possible two digit pairs. Finds the equipartition ratios
 of these numbers and appends it to our list. This
 could have been written more simply with
 nested For loops but I chose not to for clarity of how this loop correctly iterates *)
```

```
For[i = 0, i < 10, i++,
  AppendTo[equipartitionSmall, f[i, 0]];
  AppendTo[equipartitionSmall, f[i, 1]];
  AppendTo[equipartitionSmall, f[i, 2]];
  AppendTo[equipartitionSmall, f[i, 3]];
  AppendTo[equipartitionSmall, f[i, 4]];
  AppendTo[equipartitionSmall, f[i, 5]];
  AppendTo[equipartitionSmall, f[i, 6]];
  AppendTo[equipartitionSmall, f[i, 7]];
  AppendTo[equipartitionSmall, f[i, 8]];
  AppendTo[equipartitionSmall, f[i, 9]];
 ];
```

```
equipartitionSmall
```

```
(* Graph the equipartitions *)
BarChart[{equipartitionSmall}]
```

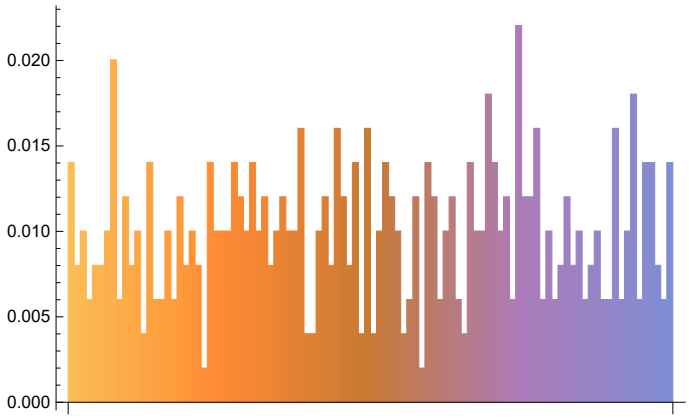


```
(* The equipartitions seem random and not
 centered around a common value. This is expected as
 we only had a small data set. The law of large numbers
 tells us that it should center on a value
 for our next two tests *)

(* Decimal expansion for the number e, medium, level 2 *)
e = RealDigits[E, 10, 500];
mediumE = e[[1]];

equipartitionMedium = {}
```

```
{}
```

```
(* This function computes the equipartition
  ratitos for a given two digits "n" and "m". Note
  how we are looping to Length[mediumE]-1 and dividing by this amount,
and not the full Length[mediumE] *)
```

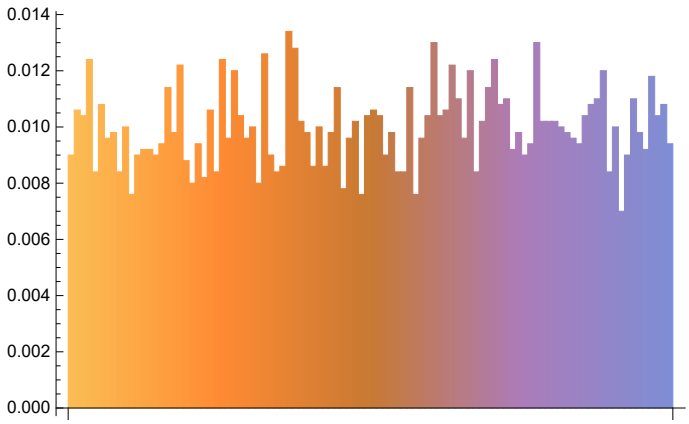
```
f[n_, m_] := N[
    Sum[If[{mediumE[[k]], mediumE[[k + 1]]} == {n, m}, 1, 0], {k, 1, Length[mediumE] - 1}] / (Length[mediumE] - 1)];

(* Loops from 00 to 99 which are all
 possible two digit pairs. Finds the equipartition ratios
 of these numbers and appends it to our list. This
 could have been written more simply with
 nested For loops but I chose not to for clarity of how this loop correctly iterates *)
For[i = 0, i < 10, i++,
  AppendTo[equipartitionMedium, f[i, 0]];
  AppendTo[equipartitionMedium, f[i, 1]];
  AppendTo[equipartitionMedium, f[i, 2]];
  AppendTo[equipartitionMedium, f[i, 3]];
  AppendTo[equipartitionMedium, f[i, 4]];
  AppendTo[equipartitionMedium, f[i, 5]];
  AppendTo[equipartitionMedium, f[i, 6]];
  AppendTo[equipartitionMedium, f[i, 7]];
  AppendTo[equipartitionMedium, f[i, 8]];
  AppendTo[equipartitionMedium, f[i, 9]];
 ];

(* Graph the equipartitions *)
BarChart[{equipartitionMedium}]
```




```
(* These values seem more centered than the smaller trial,
but they are not perfect. However, this outcome
 was expected and we expect a more perfect outcome for the large trial *)

(* Decimal expansion for the number e, large, level 2 *)
e = RealDigits[E, 10, 5000];

largeE = e[[1]];

equipartitionLarge = {}
```

```
{}
```

```
(* This function computes the equipartition
  ratitos for a given two digits "n" and "m". Note
  how we are looping to Length[largeE]-1 and dividing by this amount,
and not the full Length[largeE] *)

f[n_, m_] :=
  N[ Sum[If[{largeE[[k]], largeE[[k + 1]]} == {n, m}, 1, 0], {k, 1, Length[largeE] - 1}] / (Length[largeE] - 1)];

(* Loops from 00 to 99 which are all
 possible two digit pairs. Finds the equipartition ratios
 of these numbers and appends it to our list. This
 could have been written more simply with
 nested For loops but I chose not to for clarity of how this loop correctly iterates *)
For[i = 0, i < 10, i++,
  AppendTo[equipartitionLarge, f[i, 0]];
  AppendTo[equipartitionLarge, f[i, 1]];
  AppendTo[equipartitionLarge, f[i, 2]];
  AppendTo[equipartitionLarge, f[i, 3]];
  AppendTo[equipartitionLarge, f[i, 4]];
  AppendTo[equipartitionLarge, f[i, 5]];
  AppendTo[equipartitionLarge, f[i, 6]];
  AppendTo[equipartitionLarge, f[i, 7]];
  AppendTo[equipartitionLarge, f[i, 8]];
  AppendTo[equipartitionLarge, f[i, 9]];
 ];

(* Graph the equipartitions *)
BarChart[{equipartitionLarge}]
```




```
(* The graph seems more cented toward .01 than
  the previous graph. Since there are 100 digits between 00-99,
I'm concluding that we will need a much larger trial to see near-
 perfect equiparitions like we did in the binary
  trials. The binary trials were near perfect because for
  level 2 there are only 4 combinations of digits. In conclusion,
the number e seems to satisfy the equipartition property *)
```