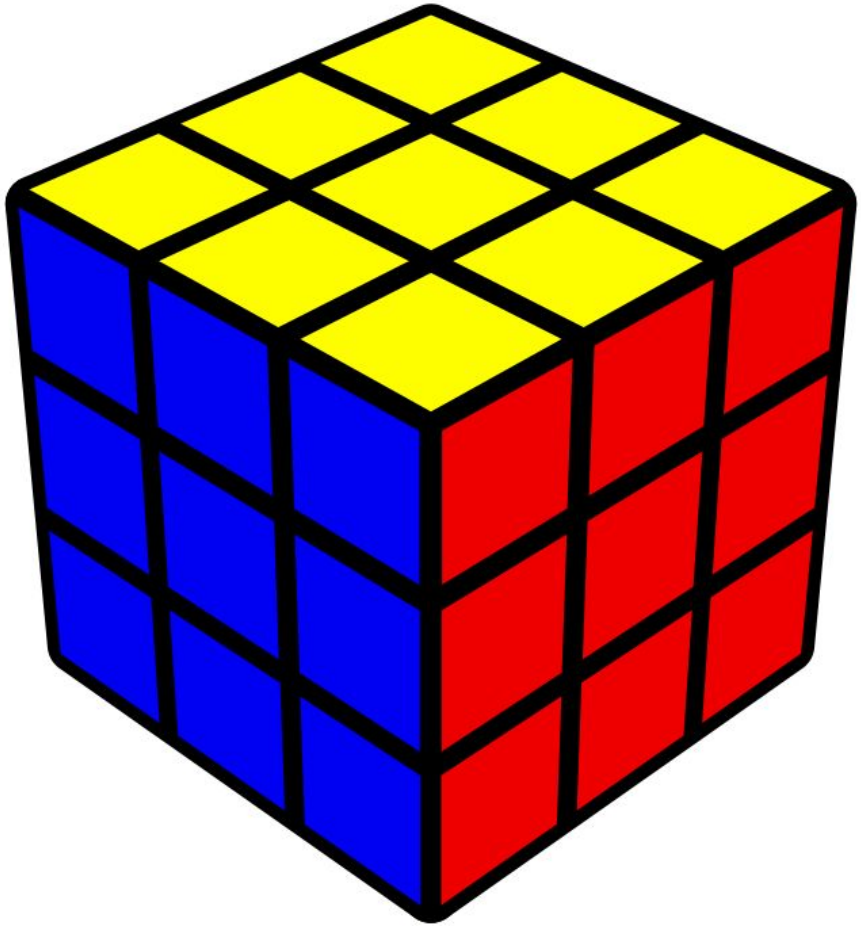
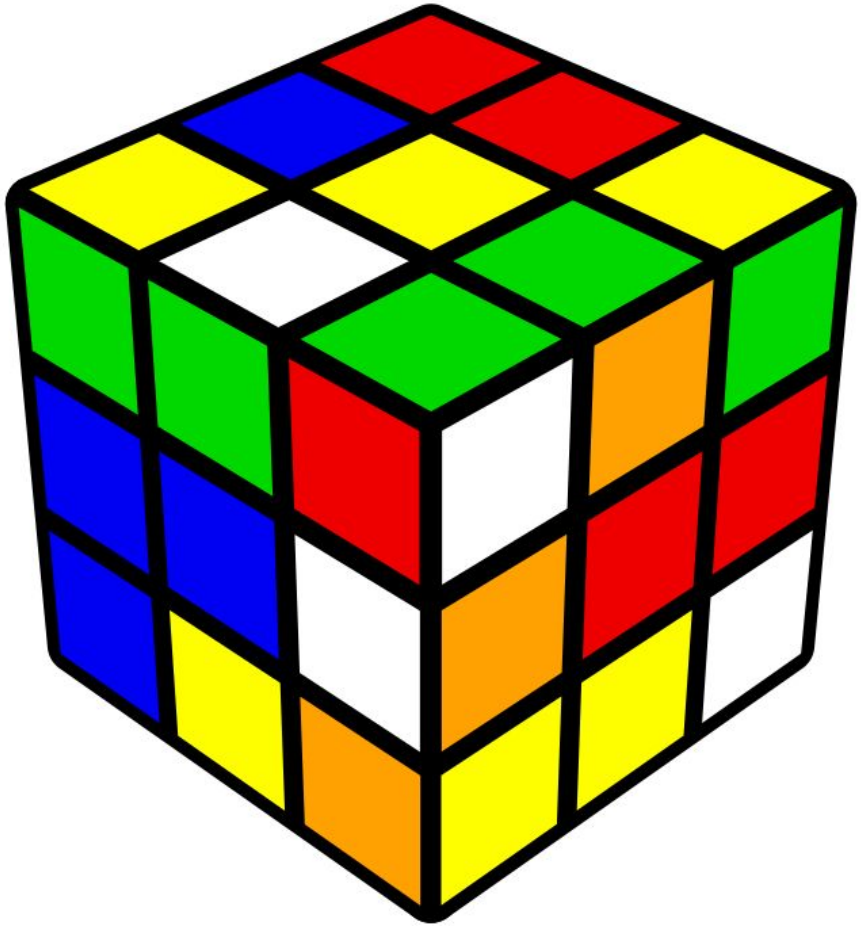




STATISTICAL ANALYSIS OF THE TIMES OF DIFFERENT RUBIK'S CUBE SCRAMBLERS

Jonathan Lam

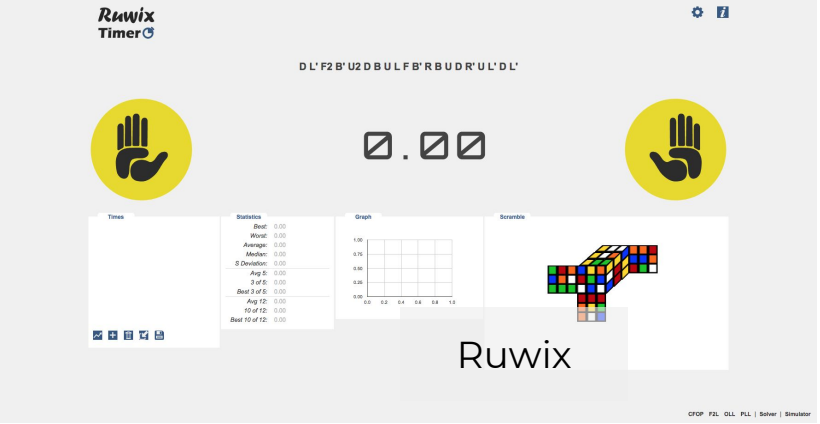
1

THE PROJECT

PROJECT OVERVIEW

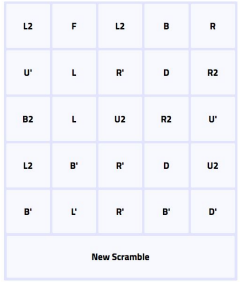
- Rubik’s cube solving is fun
- Speed-Rubik’s cube solving is more fun
- “Scramblers”: programs that provide a random scramble, time the solve, and more
- Focusing only on the scrambles -- is there a difference?

SCRAMBLERS



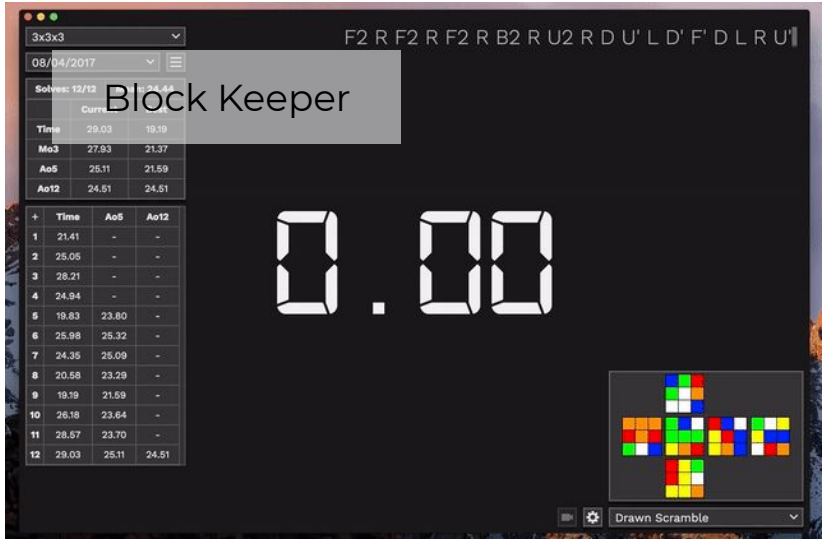
Ruwix

cubetimer.com

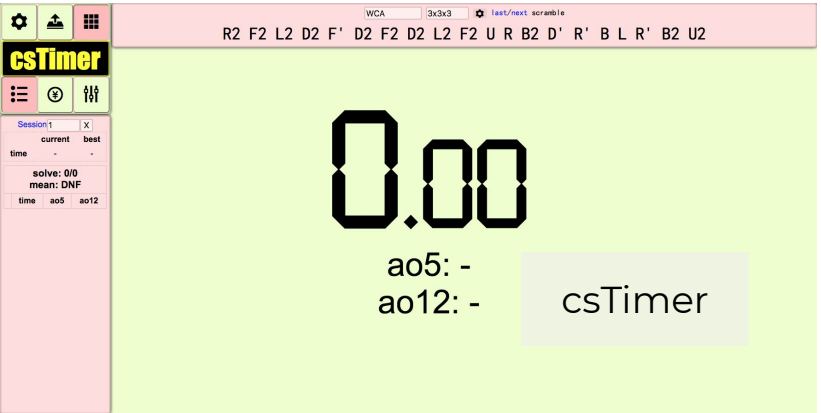


mine

qqTimer



Block Keeper



csTimer

2

THE PLAN

PROJECT PLAN

- time myself using scrambles from the scramblers
- create t-intervals for the times for every scrambler (6 tests)
- create t-tests for the difference of means between every two scramblers (15 tests)
- run a chi-square homogeneity test on the times (1 test)

DATA COLLECTION

1. generate and save 50 random scrambles from each source
2. write a program that shows me a randomly chosen scramble of all the random scrambles (but don’t show the scrambler)
3. time the scramble, and save with the others from the same scrambler
4. repeat 2 & 3 until all scrambles are exhausted

```
let fs = require('fs');
// source: https://stackoverflow.com/a/12506613
var stdin = process.stdin;
stdin.resume();
stdin.setEncoding('utf8');
// on any data into stdin
stdin.on('data', function(key) {
 // ctrl-c ( end of text )
 if(key === '\u0003') {
   process.exit();
 }
 // if 'd' (delete) is pressed in state 2, delete last scramble
 if(key === 'd\u000a' && state === 2) {
   state = 0;
   console.log('time not saved');
   generateScramble();
 }
 // when enter is pressed change state
 if(key === '\u000a') {
   switch(state) {
     case 0: startTimer(); break;
     case 1: endTimer(); break;
     case 2: saveTime(); break;
   }
   state = (state + 1) % 3;
   if(state === 0) {
     generateScramble();
   }
 }
});
// source: https://stackoverflow.com/a/34970550
function clock(start) {
 if(!start) return process.hrtime();
 var end = process.hrtime(start);
 return Math.round((end[0]*1000) + (end[1]/1000000));
}
// get data
let data = require('./scrambles.js');
// timer code
let state = 0, start, duration;
/**
 * state = 0 means waiting to start
 * state = 1 means timer running
 * state = 2 means waiting to save
 */
// start timer by beginning the duration
let startTimer = () => {
 start = clock();
 process.stdout.write('time started');
};
// end timer by getting duration and printing out the time
let endTimer = () => {
 duration = clock(start);
 process.stdout.write(duration/1000 + 's\n[d]elete? ');
};
// save time
let saveTime = () => {
 // get scramble, remove from scrambles array
 let scramble = data[scramblerIndex].scrambles.shift();
 // save scramble locally
 if(data[scramblerIndex].solved === undefined) {
   data[scramblerIndex].solved = [];
 }
 data[scramblerIndex].solved.push({
   scramble: scramble,
   duration: duration,
   date: new Date().getTime()
 });
 // save scramble in file
 fs.writeFile('./scrambles.js', `module.exports = ${ JSON.stringify(data, null, 2) };`, e => { if(e)
throw e });
};
// get a scramble and generate first scramble
let scramblerIndex;
let generateScramble;
(generateScramble = () => {
 // disregard data with zero scrambles left
 let filteredData = data.filter(scramblerData => scramblerData.scrambles.length > 0);
 // choose random scrambler
 scramblerIndex = data.map(scramblerData =>
scramblerData.name).indexOf(filteredData[Math.floor(Math.random() * filteredData.length)].name);
 // display the first scramble from the chosen scramble
 // format it so that multiple spaces are replaced with one
 process.stdout.write('\n' + data[scramblerIndex].scrambles[0].replace(/\s+/g, ' ').trim());
})();
```

REPEAT! (x300)

save time with scrambler

choose random scrambler

display *only* the scramble

THE TIMER

(the code)

Scramble: D R' B F2 R2 B2 F' R2 L U' R2 D2 B D2 R F2 L' F' R2 F L2 F B2 U' F2
Time: 15523ms

Scramble: B2 D' L' U B' F2 R' U D2 L' F' L2 D' L2 B2 R' B' F D2 F D2 L' B R2 B2
Time: 17232ms

Scramble: R2 B' F2 D2 F L2 F2 R' L F2 B D F' U2 L U F2 B2 U' F R2 D2 B2 U' F'
Time: 18951ms

(never see the scrambler name)

THE TIMER

(what I see)

3

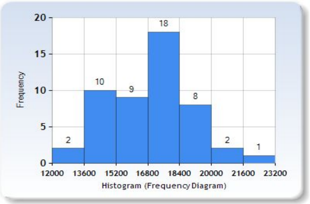
THE DATA AND MECHANICS

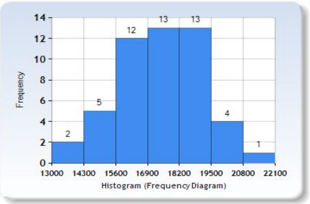
DATA!

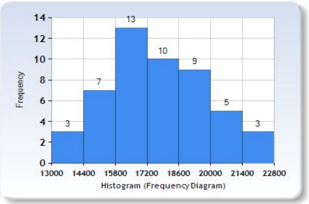
Ruwix	18508,19802,17046,12521,18110,17893,16111,17069,17811,14780,14944,15219,20841,16004,17548,19191,14331,14776,17776,17123,17984,17215,14821,14770,12088,18092,18179,13888,22141,18938,15704,16224,18208,14658,14350,18574,18810,19963,16436,14500,21465,18031,17063,19094,15328,16286,15715,17755,17998,17604
csTimer	21527,16090,16121,19067,17365,18176,16230,13369,17076,16649,18205,18043,19348,18390,18334,19616,19657,15319,16353,20489,18003,18602,17336,16294,17735,16670,18778,18555,14278,19161,17545,14921,15945,18167,16978,19787,18878,14813,19291,18950,16926,19277,17019,15470,16235,16623,15361,16139,17711,16456
cubeTimer	18051,19179,18005,14741,20726,19110,13744,17996,16158,17187,22304,17786,18730,19133,22709,15167,19869,16951,16905,15419,16529,17316,19639,16158,14825,13578,21659,15330,19667,15204,20024,20110,17446,17789,18866,16285,15482,17974,20252,20272,16958,17890,16646,16467,14249,17620,16699,19422,16272,16081
qqTimer	15729,18207,20934,21946,18151,17069,16971,15961,18737,16747,15408,12117,17421,15869,17921,16822,17264,18474,18957,19981,15945,17518,19506,18419,16927,19694,18598,15437,19704,17467,21481,15787,20975,14805,13865,19977,14706,17047,15269,14936,15469,15004,19174,17509,14299,13731,19007,14588,17806,17094
mine (not random-state)	18420,16905,16975,15965,17225,18394,17735,17775,18876,17204,16659,16195,19714,13781,20146,11272,17098,18932,16824,14985,11582,18306,18847,17926,20014,17515,17821,19039,18209,18175,18256,17568,18138,17163,18183,17417,18746,16646,19844,18196,20750,14874,16419,19052,19719,17708,17656,15323,18823,16642
Block Keeper	23602,16596,15674,17163,16388,18423,21041,16706,17744,19304,16912,14614,20829,17385,17940,19439,17417,16498,15714,21015,14795,15851,15100,14681,16667,17756,15847,16835,16131,15535,16776,16781,16811,19857,16595,19413,17219,19623,15677,20465,20264,16371,20363,18883,16008,14491,16727,20183,17227,17797

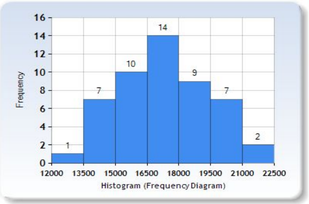
DATA!

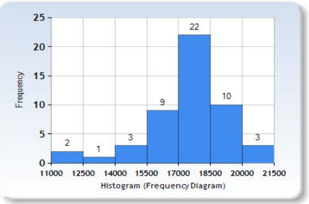
Scrambler	Mean	SD	Outliers/Gaps/Skew?
Ruwix	16985.72	2149.81	
csTimer	17466.56	1685.43	
cubeTimer	17651.58	2158.61	
qqTimer	17248.60	2174.67	
mine (not random-state)	17512.74	1880.30	three left outliers
Block Keeper	17542.66	2021.24	slight skew right

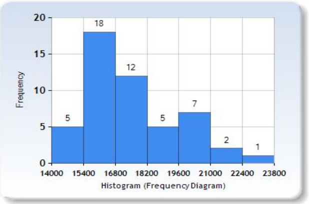












T-INTERVAL (RUWIX, 95% CONFIDENCE)

Conditions

- Independence assumption: It can be assumed that the times of different solves are mutually independent of one another.

T-INTERVAL (RUWIX, 95% CONFIDENCE)

Conditions

- Independence assumption: It can be assumed that the times of different solves are mutually independent of one another.
- Randomization condition: The scramblers were randomly assigned by a program.

T-INTERVAL (RUWIX, 95% CONFIDENCE)

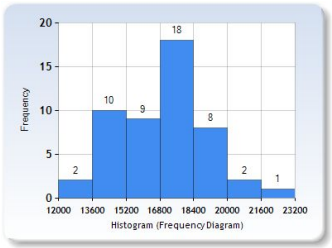
Conditions

- Independence assumption: It can be assumed that the times of different solves are mutually independent of one another.
- Randomization condition: The scramblers were randomly assigned by a program.
- 10% condition: 50 solves is less than 10% of all possible scrambles using this scrambler.

T-INTERVAL (RUWIX, 95% CONFIDENCE)

Conditions

- Independence assumption: It can be assumed that the times of different solves are mutually independent of one another.
- Randomization condition: The scramblers were randomly assigned by a program.
- 10% condition: 50 solves is less than 10% of all possible scrambles using this scrambler.
- Nearly Normal condition: A histogram of the data appears nearly Normal (unimodal and roughly symmetric), and the sample size is large.



T-INTERVAL (RUWIX, 95% CONFIDENCE)

Mechanics

$\bar{x} = 16985.72$

$df = n-1 = 49$

$t_{49}^* = 2.0095$

$SE = SD/\sqrt{(n)} = 2149.81/\sqrt{(50)} = 304.03$

$CL = \bar{x} \pm t_{49}^* \times SE = 16895.72 \pm 2.0095 \times 304.03 = (16284, 17172)$

T-INTERVAL (RUWIX, 95% CONFIDENCE)

Mechanics

$\bar{x} = 16985.72$

$df = n-1 = 49$

$t_{49}^* = 2.0095$

$SE = SD/\sqrt{(n)} = 2149.81/\sqrt{(50)} = 304.03$

$CL = \bar{x} \pm t_{49}^* \times SE = 16985.72 \pm 2.0095 \times 304.03 = (16374, 17596)$

Interpretation

We are 95% sure that the true mean for my solves using a scramble from the Ruwix scrambler is between 16.374 and 17.596 seconds.

95% CONFIDENCE T-INTERVALS (SUMMARY)

Ruwix: (16374, 17596)

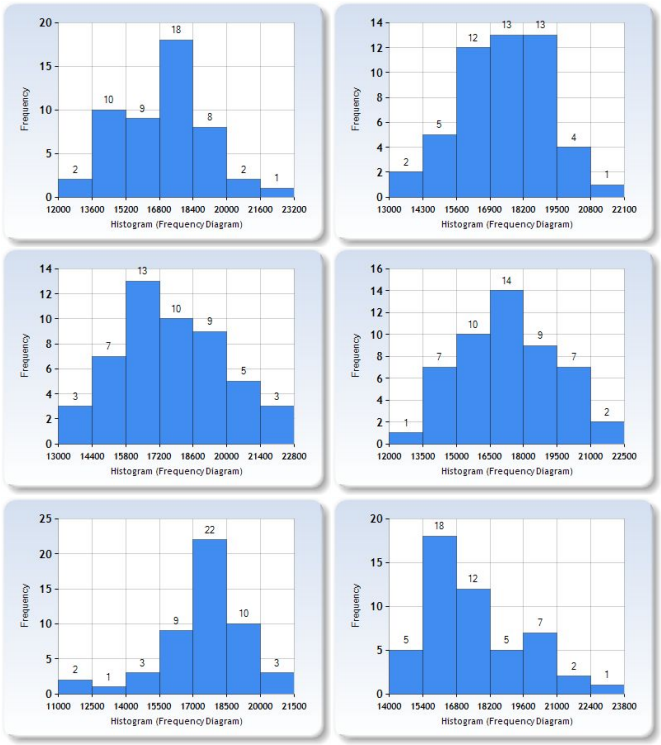
csTimer: (16988, 17946)

cubeTimer: (17038, 18265)

qqTimer: (16631, 17867)

mine: (16978, 18047)

Block Keeper: (16968, 18117)



T-TEST FOR THE DIFFERENCE OF MEANS

Hypotheses:

Null Hypothesis (H_o): There is no statistical difference between the means of the solve times of Ruwix and csTimer scrambles.

Alternate Hypothesis (H_A): There is a statistical difference between the means of the solve times of Ruwix and csTimer scrambles.

T-TEST FOR THE DIFFERENCE OF MEANS

Conditions:
Independence Assumption (already checked)
Nearly Normal Assumption (already checked)
Independent Groups Assumption: The times from one scrambler are independent of the times from another scrambler.

T-TEST FOR THE DIFFERENCE OF MEANS

Mechanics:

$\bar{x}_1 = 16985.72$; $\bar{x}_2 = 17466.56$; $\bar{x}_1 - \bar{x}_2 = -480.84$

df = 92.72 (from calculator)

$SE(x_1 - x_2) = \sqrt{(var(x_1)/n_1 + var(x_2)/n_2)} = \sqrt{(2149.81^2/50 + 1685.43^2/50)} = 386.33$

$t_{92.72} = (-480.84 - 0) / 386.33 = -1.244$

$p = P(t_{92.72} < -1.244 \cup t_{92.72} > 1.244) = 0.2164$

T-TEST FOR THE DIFFERENCE OF MEANS

Mechanics:

$\bar{x}_1 = 16985.72$; $\bar{x}_2 = 17466.56$; $\bar{x}_1 - \bar{x}_2 = -480.84$

df = 92.72 (from calculator)

$SE(x_1 - x_2) = \sqrt{(var(x_1)/n_1 + var(x_2)/n_2)} = \sqrt{(2149.81^2/50+1685.43^2/50)} = 386.33$

$t_{92.72} = (-480.84 - 0) / 386.33 = -1.244$

$p = P(t_{92.72} < -1.244 \cup t_{92.72} > 1.244) = 0.2164$

Conclusion:

Since the p-value is greater than the alpha-level of 0.05, we fail to reject the null hypothesis. Therefore, there is no evidence of a statistical difference between the means of the Ruwix and csTimer solve times.

T-TEST FOR TWO MEANS (SUMMARY)

Ruwix vs. csTimer	t = -1.245 p = 0.2164	csTimer vs. Block Keeper	t = -0.2044 p = 0.8384
Ruwix vs. cubeTimer	t = -1.545 p = 0.1255	cubeTimer vs. qqTimer	t = 0.9300 p = 0.3547
Ruwix vs. qqTimer	t = -0.6079 p = 0.5447	cubeTimer vs. mine	t = 0.3429 p = 0.7323
Ruwix vs. mine	t = -1.305 p = 0.1951	cubeTimer vs. Block Keeper	t = 0.2604 p = 0.7951
Ruwix vs. Block Keeper	t = -1.335 p = 0.1851	qqTimer vs. mine	t = -0.6497 p = 0.5174
csTimer vs. cubeTimer	t = -0.4778 p = 0.6339	qqTimer vs. Block Keeper	t = -0.7004 p = 0.4854
csTimer vs. qqTimer	t = 0.5601 p = 0.5767	mine vs. Block Keeper	t = -0.07664 p = 0.9391
csTimer vs. mine	t = -0.1293 p = 0.8974		

CHI-SQUARE TEST FOR HOMOGENEITY

Hypotheses

Null Hypothesis (H_0): There is no statistical difference between the distributions of times between different scramblers.

Alternative Hypothesis (H_A): There is a statistical difference between the distributions of times between different scramblers.

CHI-SQUARE TEST FOR HOMOGENEITY

Data (re-expressed as categories)

Time (s)	Ruwix	csTimer	cubeTimer	qqTimer	mine	Block Keeper
11–14	12	4	5	8	5	4
15	4	4	5	10	2	7
16	5	13	12	4	8	15
17	13	7	9	10	13	9
18	9	12	4	7	14	2
19–23	7	10	15	11	8	13

CHI-SQUARE TEST FOR HOMOGENEITY

Conditions:

Counted Data Condition: The data are adjusted to be counts of different categories.

Randomization Condition: The data were randomized using the experiment.

Expected Cell Frequency Condition: The data had been modified so that there are at least five expected counts in every cell. Expected frequencies:

Times	Ruwix	csTimer	cubeTimer	qqTimer	mine	Block Keeper
11-14	6.33	6.33	6.33	6.33	6.33	6.33
15	5.33	5.33	5.33	5.33	5.33	5.33
16	9.50	9.50	9.50	9.50	9.50	9.50
17	10.17	10.17	10.17	10.17	10.17	10.17
18	8.00	8.00	8.00	8.00	8.00	8.00
19-23	10.67	10.67	10.67	10.67	10.67	10.67

CHI-SQUARE TEST FOR HOMOGENEITY

Mechanics:

$\chi^2 = 46.19$

$df = (r - 1)(c - 1) = 25$

$p = P(\chi^2 > 46.19) = 0.006109$

CHI-SQUARE TEST FOR HOMOGENEITY

Mechanics:
$\chi^2 = 46.19$
$df = (r - 1)(c - 1) = 25$
$p = P(\chi^2 > 46.19) = 0.006109$

Conclusion
Because the p-value is less than the alpha-level of 0.05, we reject the null hypothesis. Therefore, we have evidence to suggest that there is a statistical difference between the distributions of the times of different scramblers.

CHI-SQUARE TEST FOR HOMOGENEITY

???

CHI-SQUARE TEST FOR HOMOGENEITY

???

- **Beware large samples.** Beware *large* samples?! That's not the advice you're used to hearing. The chi-square tests, however, are unusual. You should be wary of chi-square tests performed on very large samples. No hypothesized distribution fits perfectly, no two groups are exactly homogeneous, and two variables are rarely perfectly independent. The degrees of freedom for chi-square tests don't grow with the sample size. With a sufficiently large sample size, a chi-square test can always reject the null hypothesis. But we have no measure of how far the data are from the null model. There are no confidence intervals to help us judge the effect size.

TL;DR

We (probably) didn’t find any statistically significant results!

Which means it doesn’t matter which timer I use, because they don’t make me faster or slower.

TL;DR

We (probably) didn’t find any statistically significant results!

Which means it doesn’t matter which timer I use, because they don’t make me faster or slower.

The end. Thanks for watching.