# INFO6205 Assignment 6 Hits as time predictor()

**Ruizhe Zeng**

## 1. Program output:

```
-------Test Starts ------
Merge Sort :

ArraySize = 10000
test: StatPack {hits: 560,636, normalized=6.087; copies: 80,000, normalized=0.869;
inversions: <unset>; swaps: 95,684, normalized=1.039; fixes: 25,250,513,
normalized=274.154; compares: 184,104, normalized=1.999}
Instrumenting helper for test with 10,000 elements
time: 20.561 msec

Quick sort Dual Pivot:

ArraySize = 10000
test: StatPack {hits: 520,280, normalized=5.649; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 97,455, normalized=1.058; fixes: 26,189,665, normalized=284.351;
compares: 174,815, normalized=1.898}
Instrumenting helper for test with 10,000 elements
time: 166.41979999999998 msec

Heap Sort :

ArraySize = 10000
test: StatPack {hits: 967,204, normalized=10.501; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 124,142, normalized=1.348; fixes: 75,320,790, normalized=817.785;
compares: 235,318, normalized=2.555}
Instrumenting helper for test with 10,000 elements
time: 293.8162 msec

Merge Sort :

ArraySize = 20000
test: StatPack {hits: 1,161,458, normalized=5.864; copies: 180,000, normalized=0.909;
inversions: <unset>; swaps: 191,489, normalized=0.967; fixes: 99,636,636,
normalized=503.038; compares: 388,240, normalized=1.960}
Instrumenting helper for test with 20,000 elements
time: 6.4228 msec

Quick sort Dual Pivot:

ArraySize = 20000
```

```
test: StatPack {hits: 1,246,087, normalized=6.291; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 236,501, normalized=1.194; fixes: 108,171,240, normalized=546.127;
compares: 392,885, normalized=1.984}
Instrumenting helper for test with 20,000 elements
time: 764.8099 msec
```

Heap Sort :

```
ArraySize = 20000
test: StatPack {hits: 2,095,688, normalized=10.581; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 268,531, normalized=1.356; fixes: 303,221,739, normalized=1530.884;
compares: 510,782, normalized=2.579}
Instrumenting helper for test with 20,000 elements
time: 1230.2884999999999 msec
```

Merge Sort :

```
ArraySize = 40000
test: StatPack {hits: 2,392,040, normalized=5.643; copies: 400,000, normalized=0.944;
inversions: <unset>; swaps: 380,191, normalized=0.897; fixes: 400,054,016,
normalized=943.823; compares: 813,773, normalized=1.920}
Instrumenting helper for test with 40,000 elements
time: 13.732299999999999 msec
```

Quick sort Dual Pivot:

```
ArraySize = 40000
test: StatPack {hits: 2,530,445, normalized=5.970; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 451,354, normalized=1.065; fixes: 561,971,113, normalized=1325.824;
compares: 904,738, normalized=2.134}
Instrumenting helper for test with 40,000 elements
time: 6100.3555 msec
```

Heap Sort :

```
ArraySize = 40000
test: StatPack {hits: 4,510,400, normalized=10.641; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 576,741, normalized=1.361; fixes: 1,213,112,303, normalized=2862.023;
compares: 1,101,718, normalized=2.599}
Instrumenting helper for test with 40,000 elements
time: 5137.1852 msec
```

Merge Sort :

```
ArraySize = 80000
test: StatPack {hits: 4,944,890, normalized=5.475; copies: 880,000, normalized=0.974;
inversions: <unset>; swaps: 760,598, normalized=0.842; fixes: 1,598,730,732,
normalized=1770.108; compares: 1,707,808, normalized=1.891}
```

```
Instrumenting helper for test with 80,000 elements
time: 55.0809 msec

Quick sort Dual Pivot:

ArraySize = 80000
test: StatPack {hits: 5,072,205, normalized=5.616; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 932,859, normalized=1.033; fixes: 1,765,761,248, normalized=1955.044;
compares: 1,704,764, normalized=1.888}
Instrumenting helper for test with 80,000 elements
time: 16042.4196 msec

Heap Sort :

ArraySize = 80000
test: StatPack {hits: 9,661,322, normalized=10.697; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 1,233,853, normalized=1.366; fixes: 552,217,317, normalized=611.413;
compares: 2,362,955, normalized=2.616}
Instrumenting helper for test with 80,000 elements
time: 22518.6483 msec

Merge Sort :

ArraySize = 160000
test: StatPack {hits: 10,220,414, normalized=5.331; copies: 1,920,000, normalized=1.001;
inversions: <unset>; swaps: 1,523,744, normalized=0.795; fixes: 2,104,247,907,
normalized=1097.524; compares: 3,578,458, normalized=1.866}
Instrumenting helper for test with 160,000 elements
time: 43.8137 msec

Quick sort Dual Pivot:

ArraySize = 160000
test: StatPack {hits: 10,914,490, normalized=5.693; copies: 0, normalized=0.000; inversions:
<unset>; swaps: 1,967,668, normalized=1.026; fixes: -1,805,863,067, normalized=-941.894;
compares: 3,765,183, normalized=1.964}
Instrumenting helper for test with 160,000 elements
time: 47362.7533 msec

Heap Sort :

ArraySize = 160000
test: StatPack {hits: 20,599,822, normalized=10.744; copies: 0, normalized=0.000;
inversions: <unset>; swaps: 2,627,062, normalized=1.370; fixes: -2,090,718,764,
normalized=-1090.467; compares: 5,045,787, normalized=2.632}
Instrumenting helper for test with 160,000 elements
time: 99489.671 msec
```

```
-------Test Starts ------
Merge Sort :

ArraySize = 10000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 10,000 elements
time: 1.615429 msec

Quick sort Dual Pivot:

ArraySize = 10000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 10,000 elements
time: 1.279114 msec

Heap Sort :

ArraySize = 10000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 10,000 elements
time: 1.819424 msec

Merge Sort :

ArraySize = 20000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 20,000 elements
time: 3.612575 msec

Quick sort Dual Pivot:

ArraySize = 20000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 20,000 elements
time: 3.199251 msec

Heap Sort :

ArraySize = 20000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 20,000 elements
time: 4.181323 msec
```

Merge Sort :

```
ArraySize = 40000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes: <unset>; compares: <unset>}
Instrumenting helper for test with 40,000 elements
time: 7.803623 msec
```

Quick sort Dual Pivot:

```
ArraySize = 40000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes: <unset>; compares: <unset>}
Instrumenting helper for test with 40,000 elements
time: 6.49293 msec
```

Heap Sort :

```
ArraySize = 40000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes: <unset>; compares: <unset>}
Instrumenting helper for test with 40,000 elements
time: 9.622699 msec
```

Merge Sort :

```
ArraySize = 80000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes: <unset>; compares: <unset>}
Instrumenting helper for test with 80,000 elements
time: 17.659162 msec
```

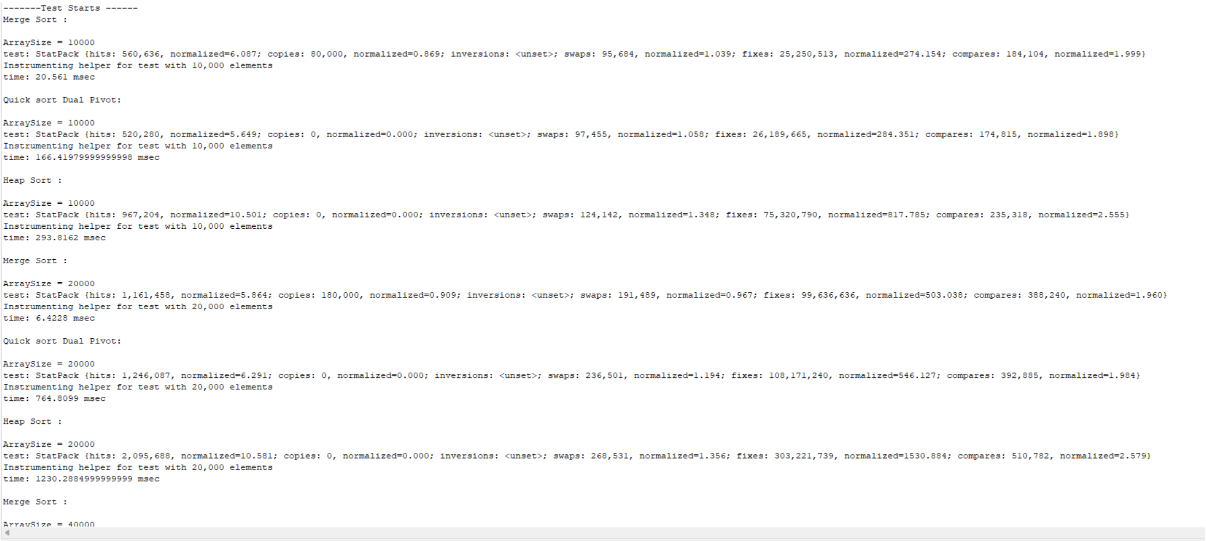
Quick sort Dual Pivot:

```
ArraySize = 80000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes: <unset>; compares: <unset>}
Instrumenting helper for test with 80,000 elements
time: 13.446567 msec
```

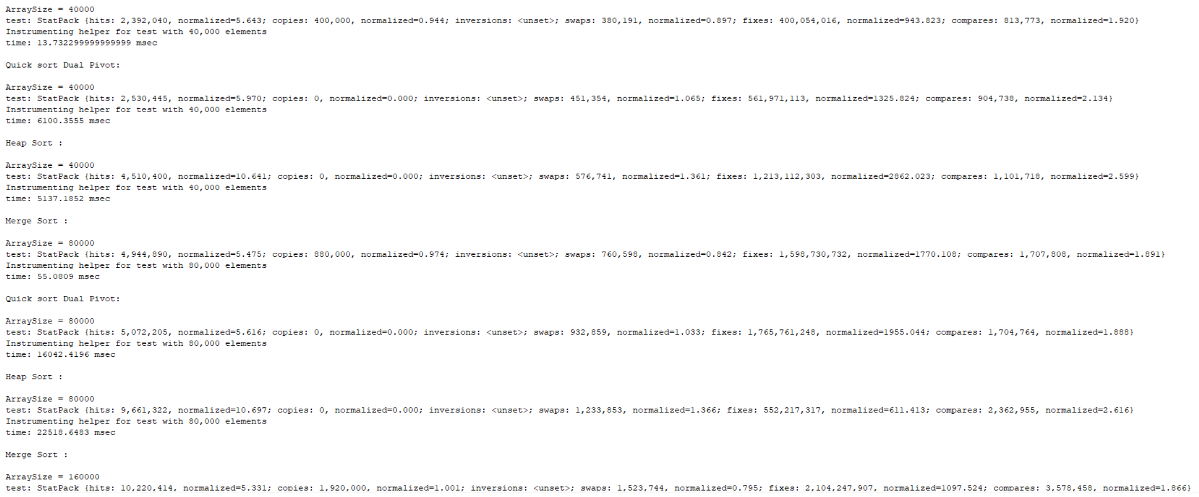
Heap Sort :

```
ArraySize = 80000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes: <unset>; compares: <unset>}
Instrumenting helper for test with 80,000 elements
time: 21.928345999999998 msec
```

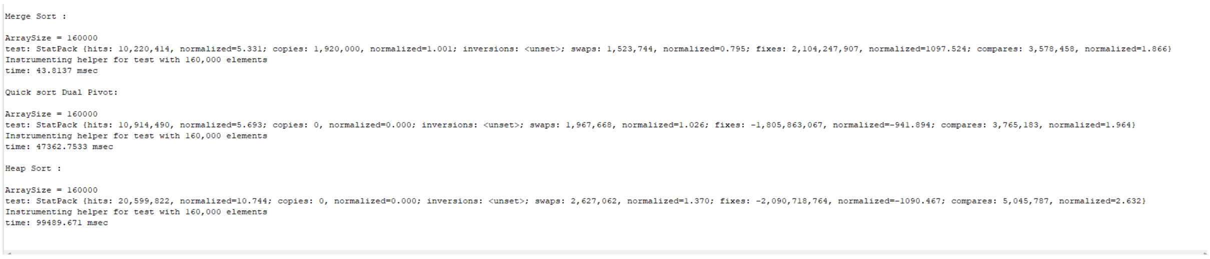
```
Merge Sort :

ArraySize = 160000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 160,000 elements
time: 40.072057 msec

Quick sort Dual Pivot:

ArraySize = 160000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 160,000 elements
time: 29.770683999999996 msec

Heap Sort :

ArraySize = 160000
test: StatPack {hits: <unset>; copies: <unset>; inversions: <unset>; swaps: <unset>; fixes:
<unset>; compares: <unset>}
Instrumenting helper for test with 160,000 elements
time: 50.429835 msec
```



```
-------Test Starts ------
Merge Sort :

ArraySize = 10000
test: StatPack {hits: 560,636, normalized=6.087; copies: 80,000, normalized=0.869; inversions: <unset>; swaps: 95,684, normalized=1.039; fixes: 25,250,513, normalized=274.154; compares: 184,104, normalized=1.999}
Instrumenting helper for test with 10,000 elements
time: 20.561 msec

Quick sort Dual Pivot:

ArraySize = 10000
test: StatPack {hits: 520,280, normalized=5.649; copies: 0, normalized=0.000; inversions: <unset>; swaps: 97,455, normalized=1.058; fixes: 26,189,665, normalized=284.351; compares: 174,815, normalized=1.898}
Instrumenting helper for test with 10,000 elements
time: 166.41979999999998 msec

Heap Sort :

ArraySize = 10000
test: StatPack {hits: 967,204, normalized=10.501; copies: 0, normalized=0.000; inversions: <unset>; swaps: 124,142, normalized=1.348; fixes: 75,320,790, normalized=817.785; compares: 235,318, normalized=2.555}
Instrumenting helper for test with 10,000 elements
time: 293.8162 msec

Merge Sort :

ArraySize = 20000
test: StatPack {hits: 1,161,458, normalized=5.864; copies: 180,000, normalized=0.909; inversions: <unset>; swaps: 191,489, normalized=0.967; fixes: 99,636,636, normalized=503.038; compares: 388,240, normalized=1.960}
Instrumenting helper for test with 20,000 elements
time: 6.4228 msec

Quick sort Dual Pivot:

ArraySize = 20000
test: StatPack {hits: 1,246,087, normalized=6.291; copies: 0, normalized=0.000; inversions: <unset>; swaps: 236,501, normalized=1.194; fixes: 108,171,240, normalized=546.127; compares: 392,885, normalized=1.984}
Instrumenting helper for test with 20,000 elements
time: 764.8099 msec

Heap Sort :

ArraySize = 20000
test: StatPack {hits: 2,095,688, normalized=10.581; copies: 0, normalized=0.000; inversions: <unset>; swaps: 268,531, normalized=1.356; fixes: 303,221,739, normalized=1530.884; compares: 510,782, normalized=2.579}
Instrumenting helper for test with 20,000 elements
time: 1230.2884999999999 msec

Merge Sort :

ArraySize = 40000
```



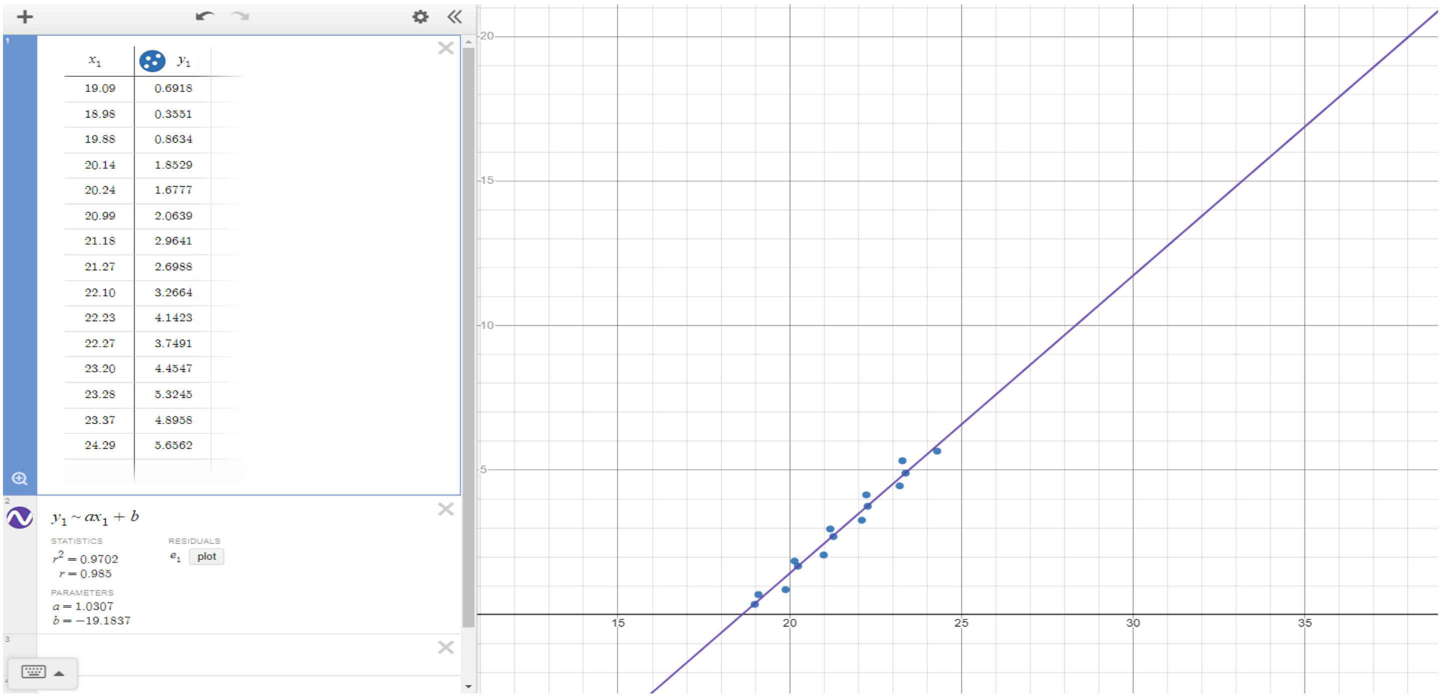
```
ArraySize = 40000
test: StatPack {hits: 2,392,040, normalized=5.643; copies: 400,000, normalized=0.944; inversions: <unset>; swaps: 380,191, normalized=0.897; fixes: 400,054,016, normalized=943.823; compares: 813,773, normalized=1.920}
Instrumenting helper for test with 40,000 elements
time: 13.732299999999999 msec

Quick sort Dual Pivot:

ArraySize = 40000
test: StatPack {hits: 2,530,445, normalized=5.970; copies: 0, normalized=0.000; inversions: <unset>; swaps: 451,354, normalized=1.065; fixes: 561,971,113, normalized=1325.824; compares: 904,738, normalized=2.134}
Instrumenting helper for test with 40,000 elements
time: 6100.3555 msec

Heap Sort :

ArraySize = 40000
test: StatPack {hits: 4,510,400, normalized=10.641; copies: 0, normalized=0.000; inversions: <unset>; swaps: 576,741, normalized=1.361; fixes: 1,213,112,303, normalized=2862.023; compares: 1,101,718, normalized=2.599}
Instrumenting helper for test with 40,000 elements
time: 5137.1852 msec

Merge Sort :

ArraySize = 80000
test: StatPack {hits: 4,944,890, normalized=5.475; copies: 880,000, normalized=0.974; inversions: <unset>; swaps: 760,598, normalized=0.842; fixes: 1,598,730,732, normalized=1770.108; compares: 1,707,808, normalized=1.891}
Instrumenting helper for test with 80,000 elements
time: 55.0809 msec

Quick sort Dual Pivot:

ArraySize = 80000
test: StatPack {hits: 5,072,205, normalized=5.616; copies: 0, normalized=0.000; inversions: <unset>; swaps: 932,859, normalized=1.033; fixes: 1,765,761,248, normalized=1955.044; compares: 1,704,764, normalized=1.888}
Instrumenting helper for test with 80,000 elements
time: 16042.4196 msec

Heap Sort :

ArraySize = 80000
test: StatPack {hits: 9,661,322, normalized=10.697; copies: 0, normalized=0.000; inversions: <unset>; swaps: 1,233,853, normalized=1.366; fixes: 552,217,317, normalized=611.413; compares: 2,362,955, normalized=2.616}
Instrumenting helper for test with 80,000 elements
time: 22518.6483 msec

Merge Sort :

ArraySize = 160000
test: StatPack {hits: 10,220,414, normalized=5.331; copies: 1,920,000, normalized=1.001; inversions: <unset>; swaps: 1,523,744, normalized=0.795; fixes: 2,104,247,907, normalized=1097.524; compares: 3,578,458, normalized=1.866}
```



```
Merge Sort :

ArraySize = 160000
test: StatPack {hits: 10,220,414, normalized=5.331; copies: 1,920,000, normalized=1.001; inversions: <unset>; swaps: 1,523,744, normalized=0.795; fixes: 2,104,247,907, normalized=1097.524; compares: 3,578,458, normalized=1.866}
Instrumenting helper for test with 160,000 elements
time: 43.8137 msec

Quick sort Dual Pivot:

ArraySize = 160000
test: StatPack {hits: 10,914,490, normalized=5.693; copies: 0, normalized=0.000; inversions: <unset>; swaps: 1,967,668, normalized=1.026; fixes: -1,805,863,067, normalized=-941.894; compares: 3,765,183, normalized=1.964}
Instrumenting helper for test with 160,000 elements
time: 47362.7533 msec

Heap Sort :

ArraySize = 160000
test: StatPack {hits: 20,599,822, normalized=10.744; copies: 0, normalized=0.000; inversions: <unset>; swaps: 2,627,062, normalized=1.370; fixes: -2,090,718,764, normalized=-1090.467; compares: 5,045,787, normalized=2.632}
Instrumenting helper for test with 160,000 elements
time: 99489.671 msec
```

## 2. Graphs:

## log(Hits) X log(time):



| $x_1$ | $y_1$ |
|---|---|
| 19.09 | 0.6918 |
| 18.98 | 0.3551 |
| 19.88 | 0.8634 |
| 20.14 | 1.8529 |
| 20.24 | 1.6777 |
| 20.99 | 2.0639 |
| 21.18 | 2.9641 |
| 21.27 | 2.6988 |
| 22.10 | 3.2664 |
| 22.23 | 4.1423 |
| 22.27 | 3.7491 |
| 23.20 | 4.4547 |
| 23.28 | 5.3245 |
| 23.37 | 4.8958 |
| 24.29 | 5.6562 |

$y_1 \sim ax_1 + b$

STATISTICS

$r^2 = 0.9702$

$r = 0.985$

RESIDUALS

$e_1$ plot

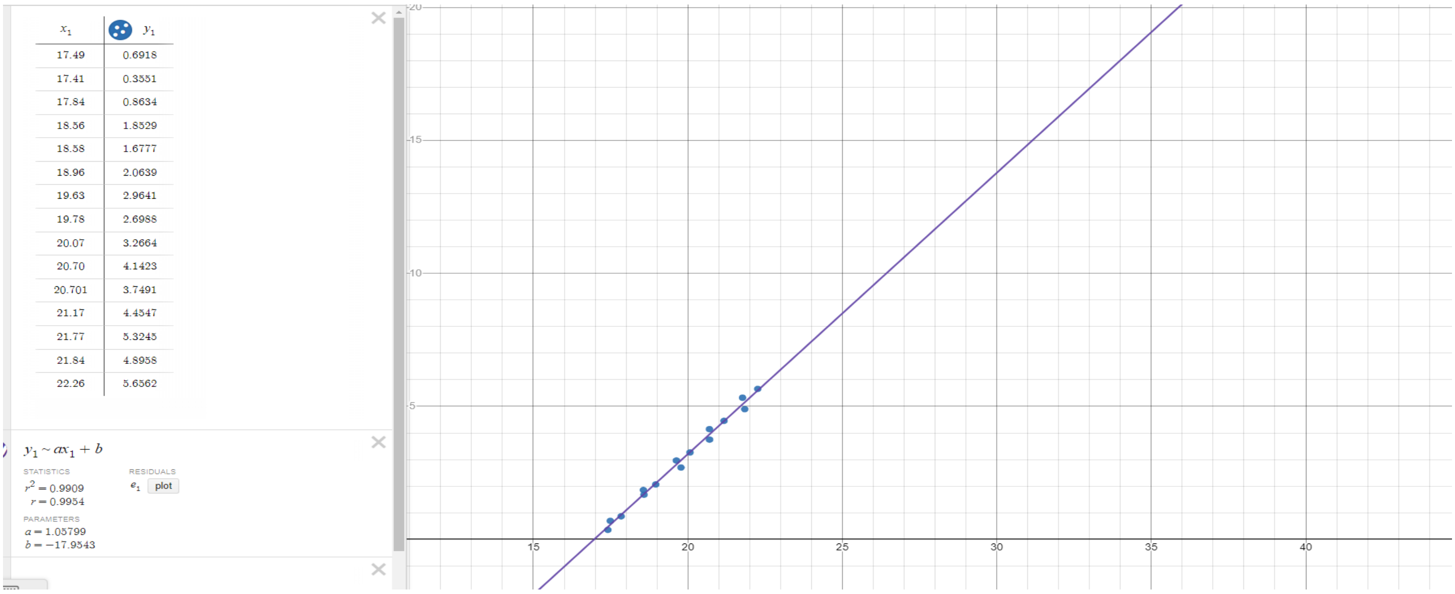
PARAMETERS

$a = 1.0307$

$b = -19.1837$

**slope = 1.0307**

## log(compare) X log(time):



| $x_1$ | $y_1$ |
|---|---|
| 17.49 | 0.6918 |
| 17.41 | 0.3551 |
| 17.84 | 0.8634 |
| 18.56 | 1.8529 |
| 18.58 | 1.6777 |
| 18.96 | 2.0639 |
| 19.63 | 2.9641 |
| 19.78 | 2.6988 |
| 20.07 | 3.2664 |
| 20.70 | 4.1423 |
| 20.701 | 3.7491 |
| 21.17 | 4.4547 |
| 21.77 | 5.3245 |
| 21.84 | 4.8958 |
| 22.26 | 5.6562 |

$y_1 \sim ax_1 + b$

STATISTICS

$r^2 = 0.9909$

$r = 0.9954$

RESIDUALS

$e_1$ plot

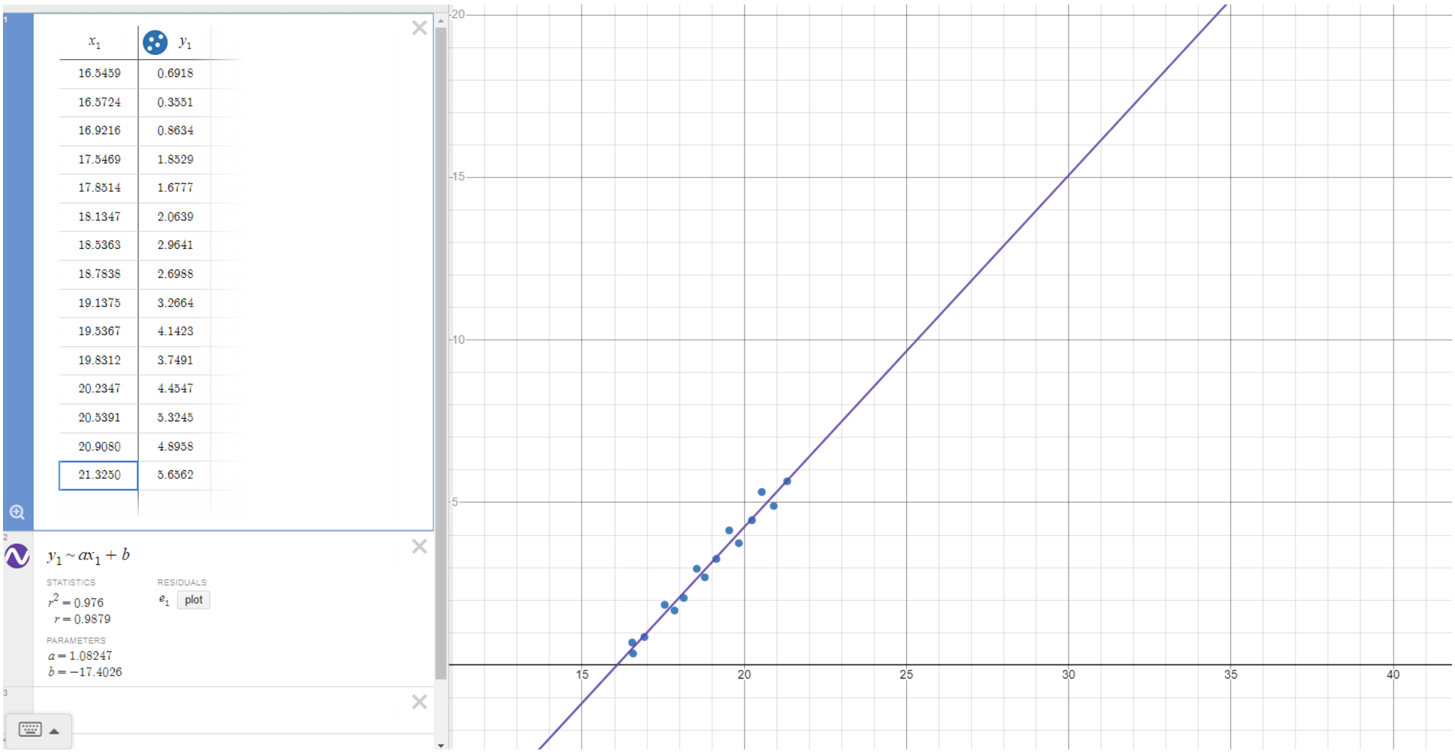
PARAMETERS

$a = 1.05799$

$b = -17.9543$

**slope = 1.0579**

## log(swap) X log(time)



| $x_1$ | $y_1$ |
|---|---|
| 16.5459 | 0.6918 |
| 16.5724 | 0.3551 |
| 16.9216 | 0.8634 |
| 17.5469 | 1.8529 |
| 17.8514 | 1.6777 |
| 18.1347 | 2.0639 |
| 18.5363 | 2.9641 |
| 18.7838 | 2.6988 |
| 19.1375 | 3.2664 |
| 19.5367 | 4.1423 |
| 19.8312 | 3.7491 |
| 20.2347 | 4.4547 |
| 20.5391 | 5.3245 |
| 20.9080 | 4.8958 |
| 21.3250 | 5.6562 |

$y_1 \sim ax_1 + b$

STATISTICS

$r^2 = 0.976$

$r = 0.9879$

RESIDUALS

$e_1$ plot

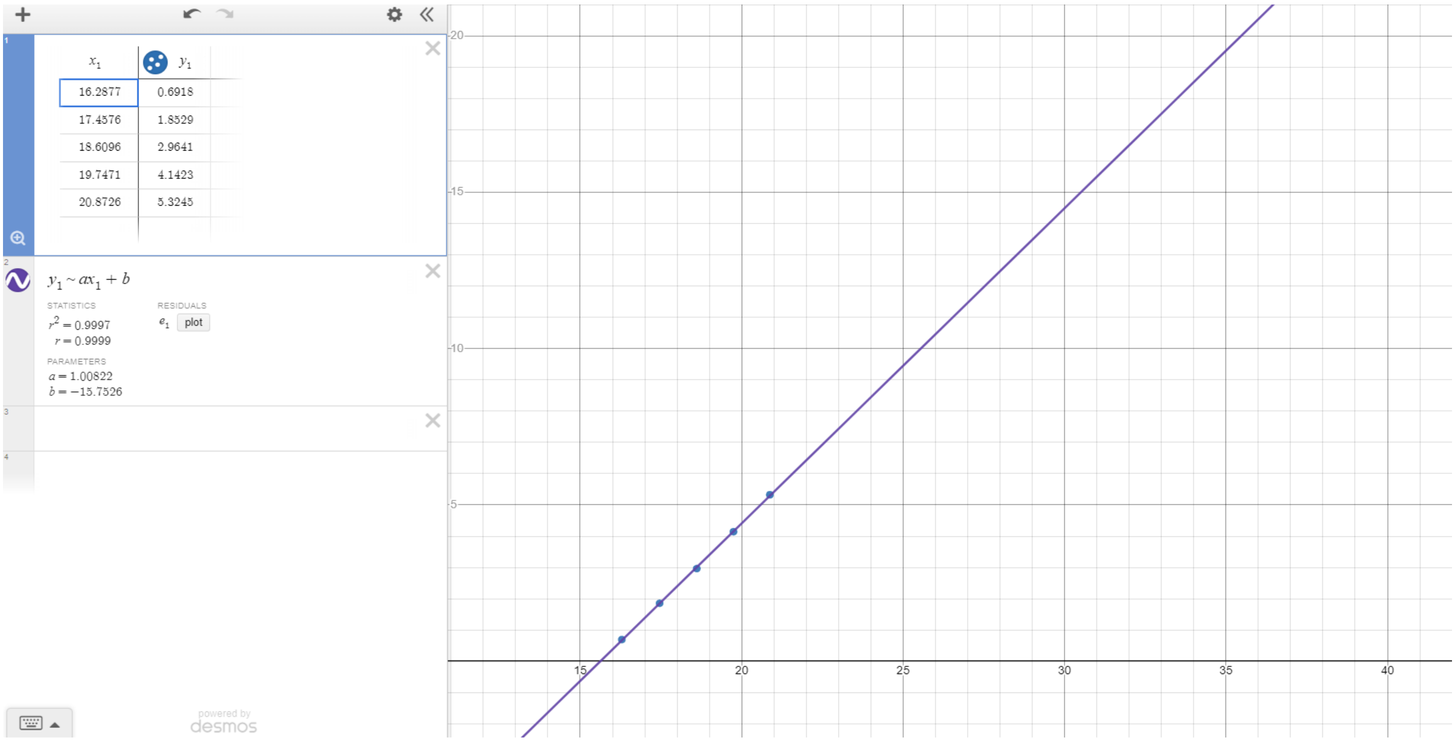
PARAMETERS

$a = 1.08247$

$b = -17.4026$

**slope = 1.0824**

**log(copy) X log(time):**



| $x_1$ | $y_1$ |
|---|---|
| 16.2877 | 0.6918 |
| 17.4576 | 1.8529 |
| 18.6096 | 2.9641 |
| 19.7471 | 4.1423 |
| 20.8726 | 5.3245 |

$y_1 \sim ax_1 + b$

STATISTICS

$r^2 = 0.9997$

$r = 0.9999$

RESIDUALS

$e_1$ plot

PARAMETERS

$a = 1.00822$

$b = -15.7526$

**slope = 1.0082**

## 3. Conclution:

**Best predictor of total execution time is still the length of the input array. Besides that base on the graphs, log(swap) X log(time) graph has the greatest slope which means swap has greater influence on the total execution time among other options,so the num of swap is also a very good predictor of total time.**

**swap > compare > hit > copy**