

Program 2 Graph Analysis

Ryan Schaefer and Wes Anderson

Create Dataset

```
library(ggplot2)
library(ggpubr)
data = read.csv("WesDataRun1.csv")
data$n2 = data$size ^ 2
data$nlogn = log(data$size) * data$size
data
```

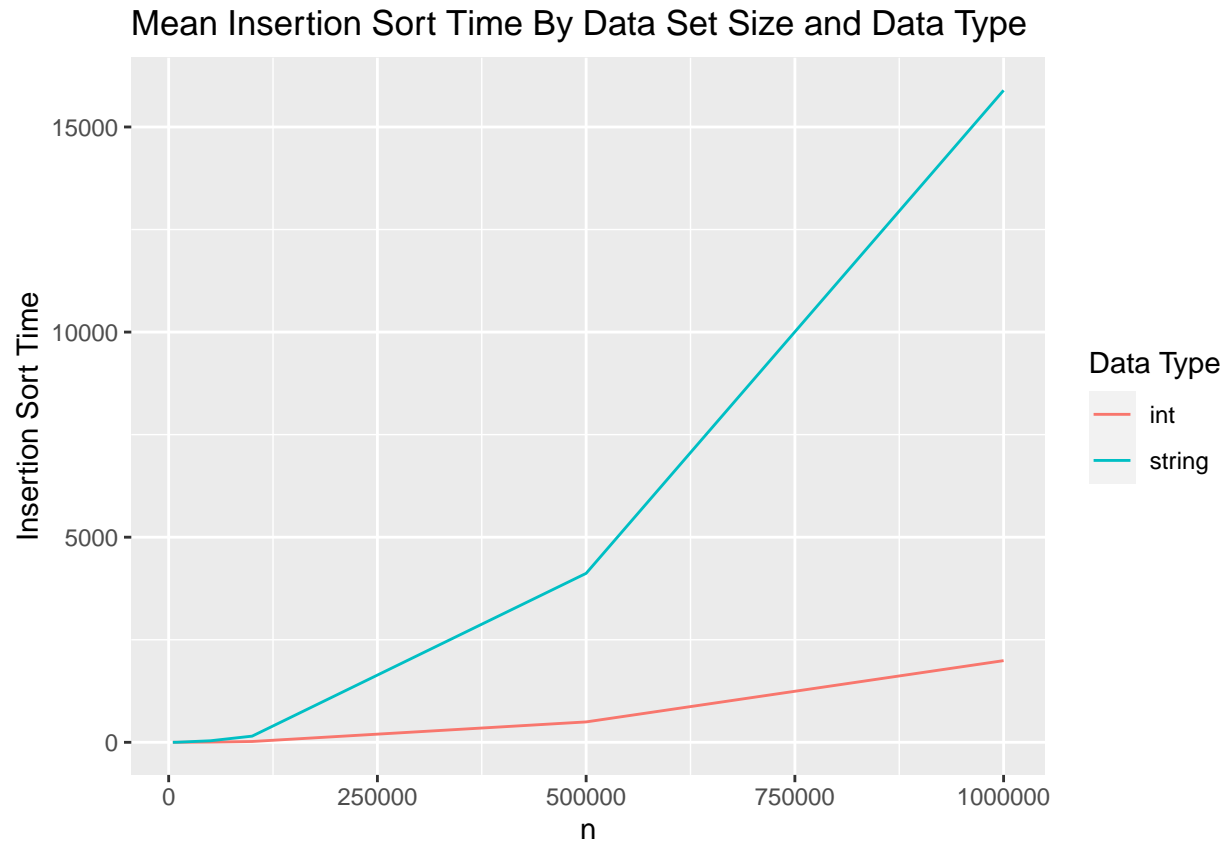
##	var_type	size	format	insertion_time	quick_time	merge_time
## 1	int	500000	noDuplicates	7.95005e+02	0.16505100	2.1168700
## 2	int	1000000	40duplicates	3.16407e+03	0.32676400	4.2730800
## 3	int	100000	40duplicates	3.15688e+01	0.03059860	0.4175630
## 4	int	10000	40duplicates	3.32033e-01	0.00240456	0.0348771
## 5	int	50000	sorted	6.90529e-04	0.01030740	0.1977440
## 6	int	50000	20duplicates	7.85778e+00	0.01516270	0.2023640
## 7	int	5000	noDuplicates	8.12401e-02	0.00139387	0.0193010
## 8	int	500000	sorted	7.13136e-03	0.09585230	2.0201300
## 9	int	500000	60sorted	1.26520e+02	0.11619200	2.0642600
## 10	int	10000	60sorted	4.75302e-02	0.00199900	0.0372471
## 11	int	1000000	noDuplicates	3.13320e+03	0.31941900	4.3305400
## 12	int	1000000	20duplicates	3.16243e+03	0.34211100	4.3177300
## 13	int	50000	noDuplicates	7.92707e+00	0.01443500	0.2101600
## 14	int	5000	60sorted	1.33967e-02	0.00132197	0.0204084
## 15	int	5000	sorted	6.16390e-05	0.00101474	0.0166772
## 16	int	100000	20duplicates	3.16373e+01	0.02955740	0.4144030
## 17	int	50000	60sorted	1.26396e+00	0.01188120	0.2052760
## 18	int	10000	noDuplicates	3.23680e-01	0.00291363	0.0396229
## 19	int	500000	20duplicates	7.86741e+02	0.16625900	2.0826600
## 20	int	500000	40duplicates	7.76092e+02	0.15903700	2.0506100
## 21	int	1000000	sorted	1.22738e-02	0.19862100	4.0315300
## 22	int	5000	20duplicates	7.45130e-02	0.00128955	0.0180706
## 23	int	100000	noDuplicates	3.10339e+01	0.02829320	0.4001210
## 24	int	50000	40duplicates	7.75182e+00	0.01533210	0.2010430
## 25	int	10000	20duplicates	3.18130e-01	0.00289081	0.0402507
## 26	int	100000	sorted	1.48089e-03	0.02122200	0.3884380
## 27	int	100000	60sorted	5.00708e+00	0.02511440	0.4058680
## 28	int	10000	sorted	1.48140e-04	0.00184614	0.0396463
## 29	int	5000	40duplicates	8.49797e-02	0.00135574	0.0203961
## 30	int	1000000	60sorted	4.96103e+02	0.24900200	4.0825000
## 31	string	50000	sorted	5.34647e-03	0.07387960	0.2638780
## 32	string	500000	20duplicates	5.14512e+03	0.97943700	3.3090600
## 33	string	50000	20duplicates	5.10532e+01	0.07543130	0.2956520

## 34	string	10000	40duplicates	2.04046e+00	0.01292500	0.0572353
## 35	string	10000	60sorted	1.29043e+00	0.01346670	0.0556319
## 36	string	100000	sorted	1.10993e-02	0.16852600	0.5636770
## 37	string	5000	40duplicates	5.10891e-01	0.00649668	0.0261124
## 38	string	500000	60sorted	3.28954e+03	0.93865900	3.0265600
## 39	string	50000	noDuplicates	5.16270e+01	0.07948510	0.3091600
## 40	string	500000	40duplicates	5.14260e+03	1.01228000	3.2950600
## 41	string	5000	20duplicates	5.16617e-01	0.00653917	0.0287966
## 42	string	100000	noDuplicates	2.05698e+02	0.16876200	0.6321520
## 43	string	5000	noDuplicates	5.21201e-01	0.00642101	0.0253947
## 44	string	100000	60sorted	1.31478e+02	0.16656500	0.5803110
## 45	string	1000000	20duplicates	2.09586e+04	2.14006000	6.8582900
## 46	string	10000	noDuplicates	2.05775e+00	0.01611400	0.0544607
## 47	string	1000000	noDuplicates	2.34202e+04	3.70442000	11.4343000
## 48	string	1000000	sorted	1.46393e-01	2.64701000	11.9568000
## 49	string	500000	noDuplicates	7.01488e+03	1.00256000	3.3534600
## 50	string	100000	40duplicates	2.09063e+02	0.19067700	0.6149120
## 51	string	5000	60sorted	3.25228e-01	0.00628677	0.0261034
## 52	string	1000000	60sorted	1.41741e+04	2.04694000	6.3251100
## 53	string	5000	sorted	4.50962e-04	0.00536470	0.0235652
## 54	string	100000	20duplicates	2.05982e+02	0.17654600	0.6268760
## 55	string	10000	20duplicates	2.03632e+00	0.01560860	0.0614365
## 56	string	10000	sorted	9.79634e-04	0.01251130	0.0522329
## 57	string	500000	sorted	4.75168e-02	0.95233200	2.9277300
## 58	string	50000	40duplicates	5.13650e+01	0.07892800	0.3075030
## 59	string	50000	60sorted	3.32685e+01	0.07817330	0.2773730
## 60	string	1000000	40duplicates	2.09124e+04	2.06010000	6.7977200
##	shell_time	intro_time	tim_time	n2	nlogn	
## 1	0.265615000	0.67225900	0.74547300	2.5e+11	6561181.69	
## 2	0.592526000	1.45301000	1.58412000	1.0e+12	13815510.56	
## 3	0.040778200	0.12452300	0.13062500	1.0e+10	1151292.55	
## 4	0.002582330	0.00961603	0.01038150	1.0e+08	92103.40	
## 5	0.005058930	0.05164430	0.05012550	2.5e+09	540988.91	
## 6	0.018773600	0.06001270	0.06192480	2.5e+09	540988.91	
## 7	0.001326450	0.00495500	0.00505642	2.5e+07	42585.97	
## 8	0.065695800	0.64501200	0.59772900	2.5e+11	6561181.69	
## 9	0.137265000	0.63992100	0.64383600	2.5e+11	6561181.69	
## 10	0.001449060	0.00919040	0.00962555	1.0e+08	92103.40	
## 11	0.591718000	1.43119000	1.55271000	1.0e+12	13815510.56	
## 12	0.595075000	1.47730000	1.54371000	1.0e+12	13815510.56	
## 13	0.019980500	0.06349480	0.06460700	2.5e+09	540988.91	
## 14	0.000664547	0.00491135	0.00414646	2.5e+07	42585.97	
## 15	0.000342053	0.00454191	0.00404962	2.5e+07	42585.97	
## 16	0.041880100	0.12514300	0.13178000	1.0e+10	1151292.55	
## 17	0.010177800	0.05137300	0.05354640	2.5e+09	540988.91	
## 18	0.002805930	0.01047000	0.01082610	1.0e+08	92103.40	
## 19	0.276191000	0.68597700	0.72346400	2.5e+11	6561181.69	
## 20	0.255003000	0.68750300	0.71473900	2.5e+11	6561181.69	
## 21	0.132546000	1.27628000	1.25504000	1.0e+12	13815510.56	
## 22	0.001103840	0.00443419	0.00477446	2.5e+07	42585.97	
## 23	0.046334700	0.11598900	0.14238000	1.0e+10	1151292.55	
## 24	0.018745300	0.06389830	0.06737100	2.5e+09	540988.91	
## 25	0.002934560	0.01055090	0.01119010	1.0e+08	92103.40	
## 26	0.009435880	0.11134500	0.10641200	1.0e+10	1151292.55	

```
## 27 0.023200100 0.11279400 0.10789600 1.0e+10 1151292.55
## 28 0.000886534 0.00933530 0.00950768 1.0e+08 92103.40
## 29 0.001372570 0.00518506 0.00556664 2.5e+07 42585.97
## 30 0.281297000 1.39514000 1.37132000 1.0e+12 13815510.56
## 31 0.057227000 0.13653300 0.12684100 2.5e+09 540988.91
## 32 2.222900000 2.05527000 2.28578000 2.5e+11 6561181.69
## 33 0.163702000 0.16297800 0.17953700 2.5e+09 540988.91
## 34 0.020406200 0.02682590 0.03173440 1.0e+08 92103.40
## 35 0.022979700 0.02751490 0.02605040 1.0e+08 92103.40
## 36 0.117571000 0.32720200 0.27328800 1.0e+10 1151292.55
## 37 0.008972860 0.01203780 0.01451480 2.5e+07 42585.97
## 38 2.107770000 1.96857000 1.88870000 2.5e+11 6561181.69
## 39 0.156538000 0.16552400 0.18918100 2.5e+09 540988.91
## 40 2.519060000 2.14414000 2.30078000 2.5e+11 6561181.69
## 41 0.008434710 0.01103280 0.01411310 2.5e+07 42585.97
## 42 0.366069000 0.34577400 0.39171200 1.0e+10 1151292.55
## 43 0.009774680 0.01259800 0.01449530 2.5e+07 42585.97
## 44 0.334957000 0.33161300 0.32746600 1.0e+10 1151292.55
## 45 5.664340000 4.66392000 4.83753000 1.0e+12 13815510.56
## 46 0.024775100 0.03463140 0.03478310 1.0e+08 92103.40
## 47 7.935090000 8.12423000 8.56265000 1.0e+12 13815510.56
## 48 1.873910000 10.17380000 6.05236000 1.0e+12 13815510.56
## 49 2.362380000 2.05452000 2.42759000 2.5e+11 6561181.69
## 50 0.361969000 0.35122200 0.40865600 1.0e+10 1151292.55
## 51 0.007896310 0.01158290 0.01143420 2.5e+07 42585.97
## 52 4.680090000 4.36611000 4.01885000 1.0e+12 13815510.56
## 53 0.004536150 0.01185560 0.01047710 2.5e+07 42585.97
## 54 0.358063000 0.38033100 0.44681100 1.0e+10 1151292.55
## 55 0.020422000 0.02618030 0.03146440 1.0e+08 92103.40
## 56 0.009215910 0.02485190 0.01878570 1.0e+08 92103.40
## 57 0.721358000 1.96970000 1.66419000 2.5e+11 6561181.69
## 58 0.161091000 0.16140900 0.20064300 2.5e+09 540988.91
## 59 0.151833000 0.16607700 0.16001200 2.5e+09 540988.91
## 60 5.346170000 4.32209000 4.94691000 1.0e+12 13815510.56
```

Insertion Sort

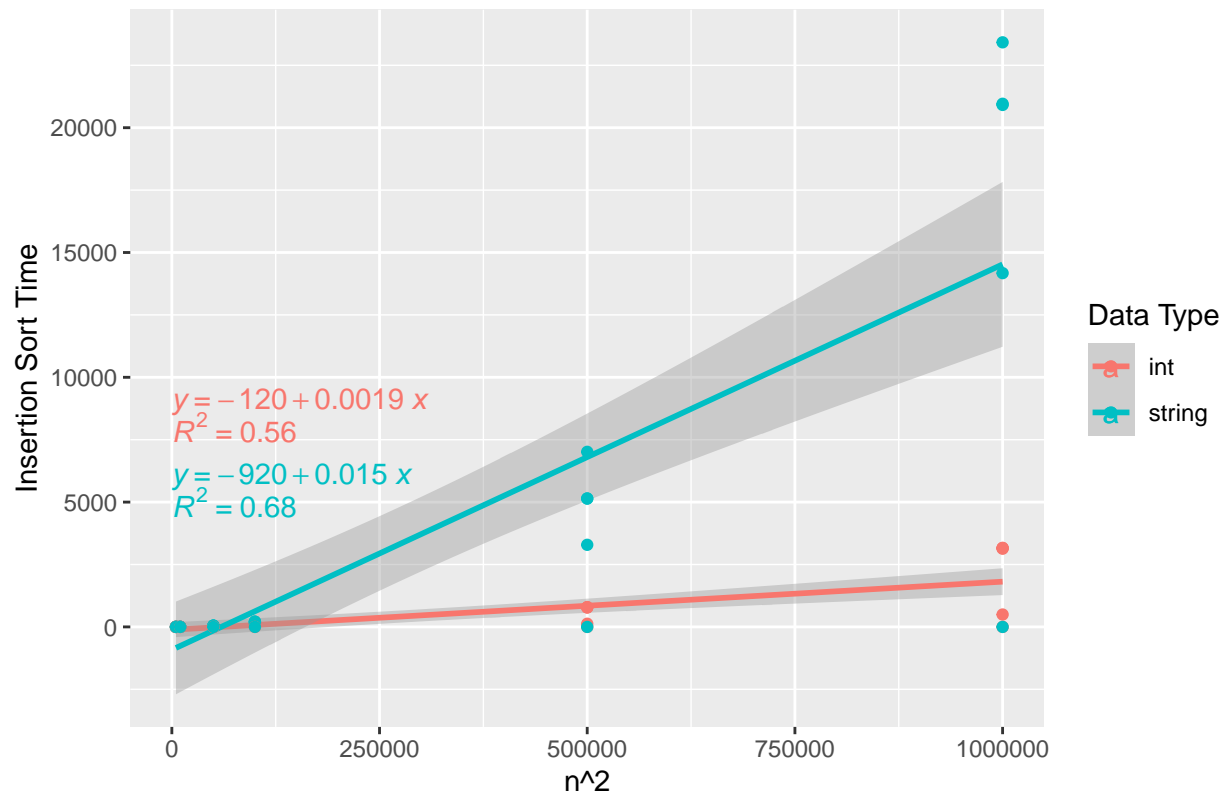
```
insertionTimes = aggregate(insertion_time ~ var_type + size + n2 + format, data = data, FUN = mean)
insertionTimes2 = aggregate(insertion_time ~ var_type + size + n2, data = data, FUN = mean)
ggplot(insertionTimes2, aes(x = size, y = insertion_time, color = var_type)) +
  geom_line() +
  labs(title = "Mean Insertion Sort Time By Data Set Size and Data Type", x = "n", y = "Insertion Sort Time")
guides(color = guide_legend(title = "Data Type"))
```



```
ggplot(insertionTimes, aes(x = size, y = insertion_time, color = var_type)) +
  labs(title = "Insertion Sort Regression Models By Data Type", x = "n^2", y = "Insertion Sort Time") +
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(9000, 6000)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(8000, 5000)) +
  guides(color = guide_legend(title = "Data Type"))
```

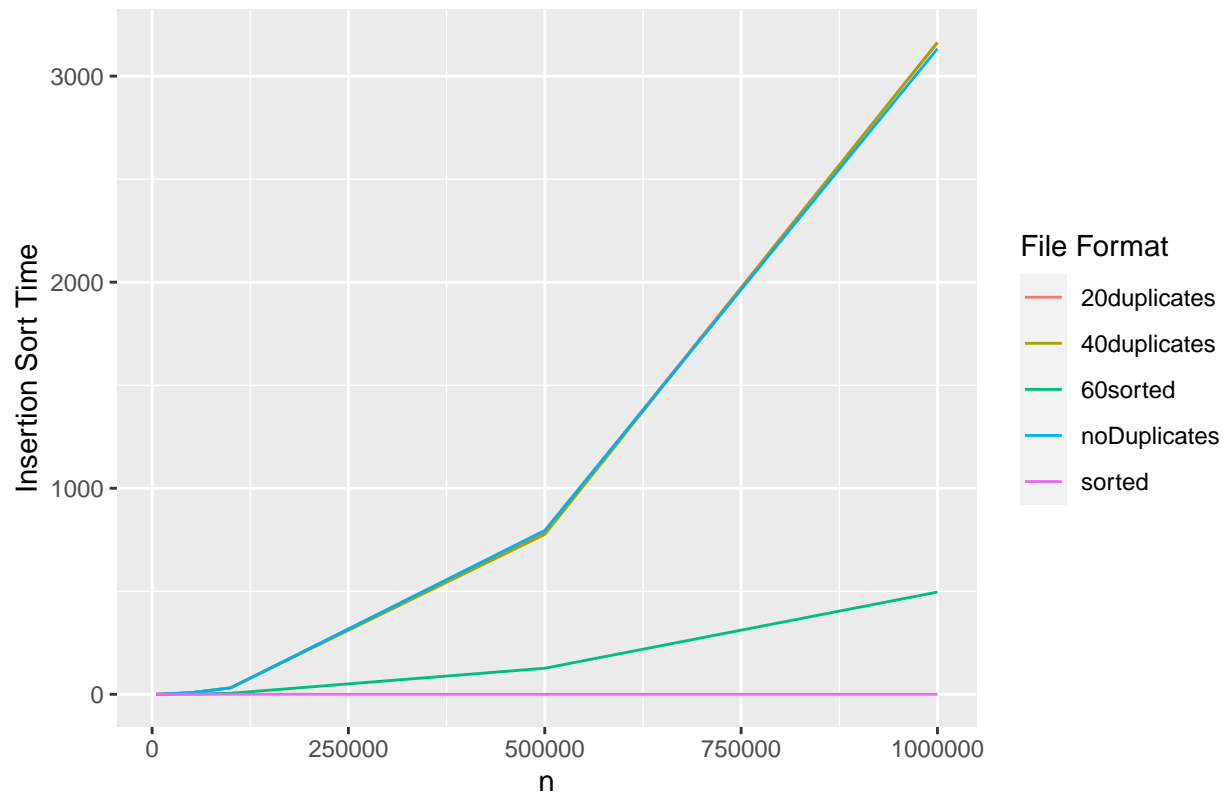
```
## 'geom_smooth()' using formula 'y ~ x'
```

Insertion Sort Regression Models By Data Type



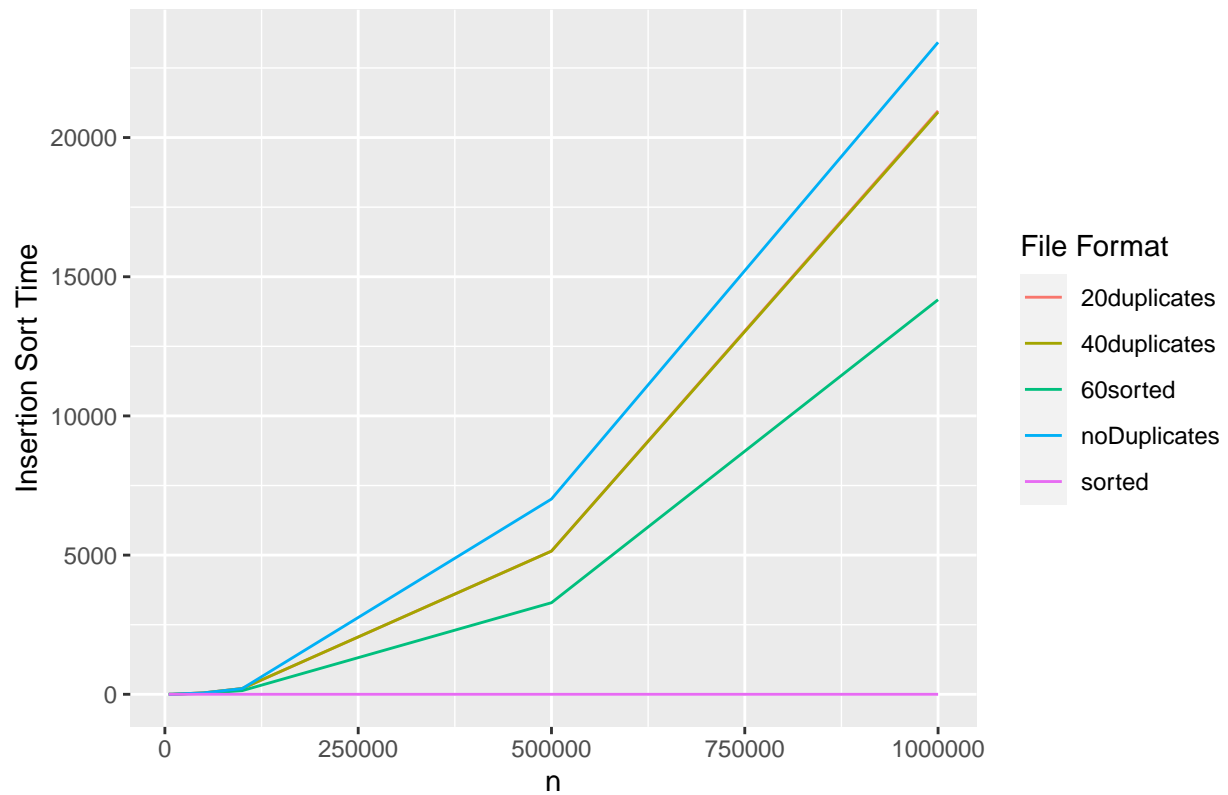
```
insertionInts = subset(insertionTimes, var_type == "int")
ggplot(insertionInts, aes(x = size, y = insertion_time, color = format)) +
  geom_line() +
  labs(title = "Insertion Sort Time With Integer Data By Data Set Size and File Format", x = "n", y = "Insertion Sort Time") +
  guides(color = guide_legend(title = "File Format"))
```

Insertion Sort Time With Integer Data By Data Set Size and File Format



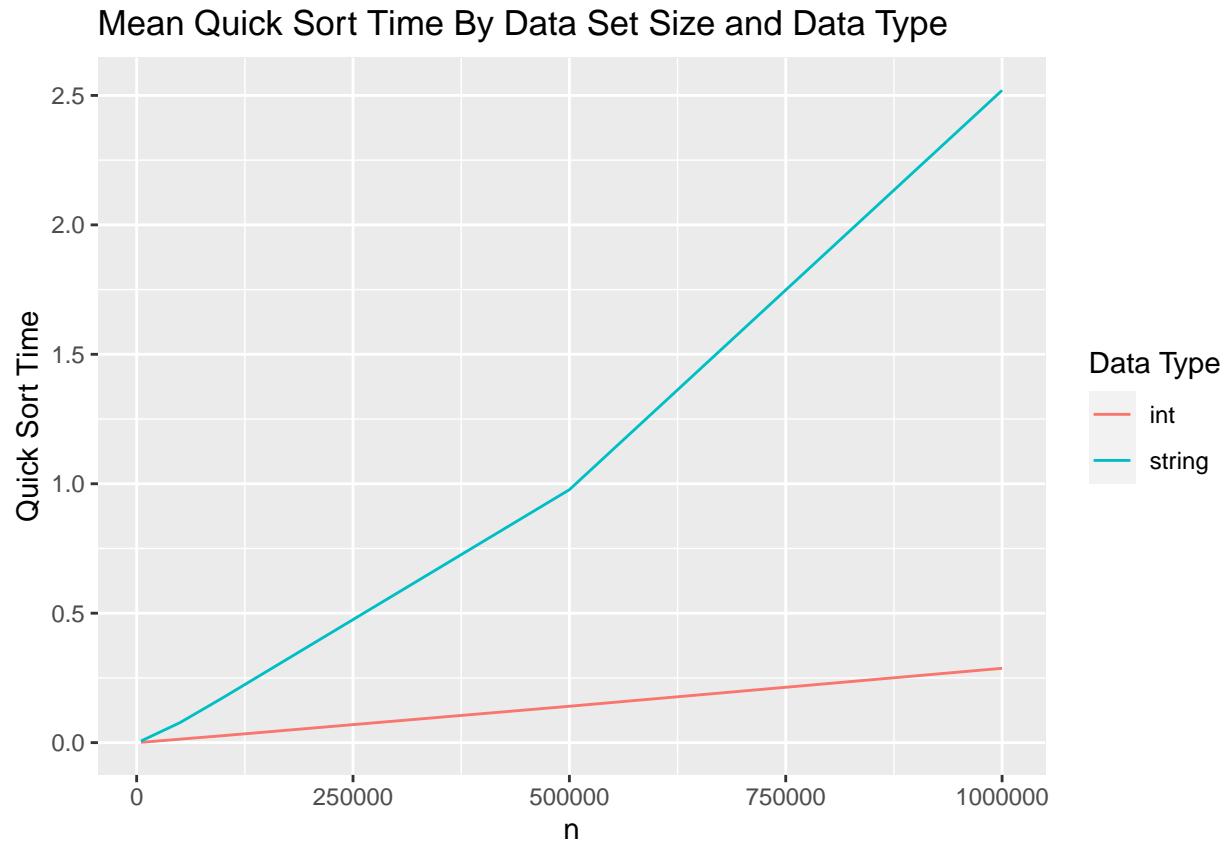
```
insertionStrings = subset(insertionTimes, var_type == "string")
ggplot(insertionStrings, aes(x = size, y = insertion_time, color = format)) +
  geom_line() +
  labs(title = "Insertion Sort Time With String Data By Data Set Size and File Format", x = "n", y = "Insertion Sort Time") +
  guides(color = guide_legend(title = "File Format"))
```

Insertion Sort Time With String Data By Data Set Size and File Format



Quick Sort

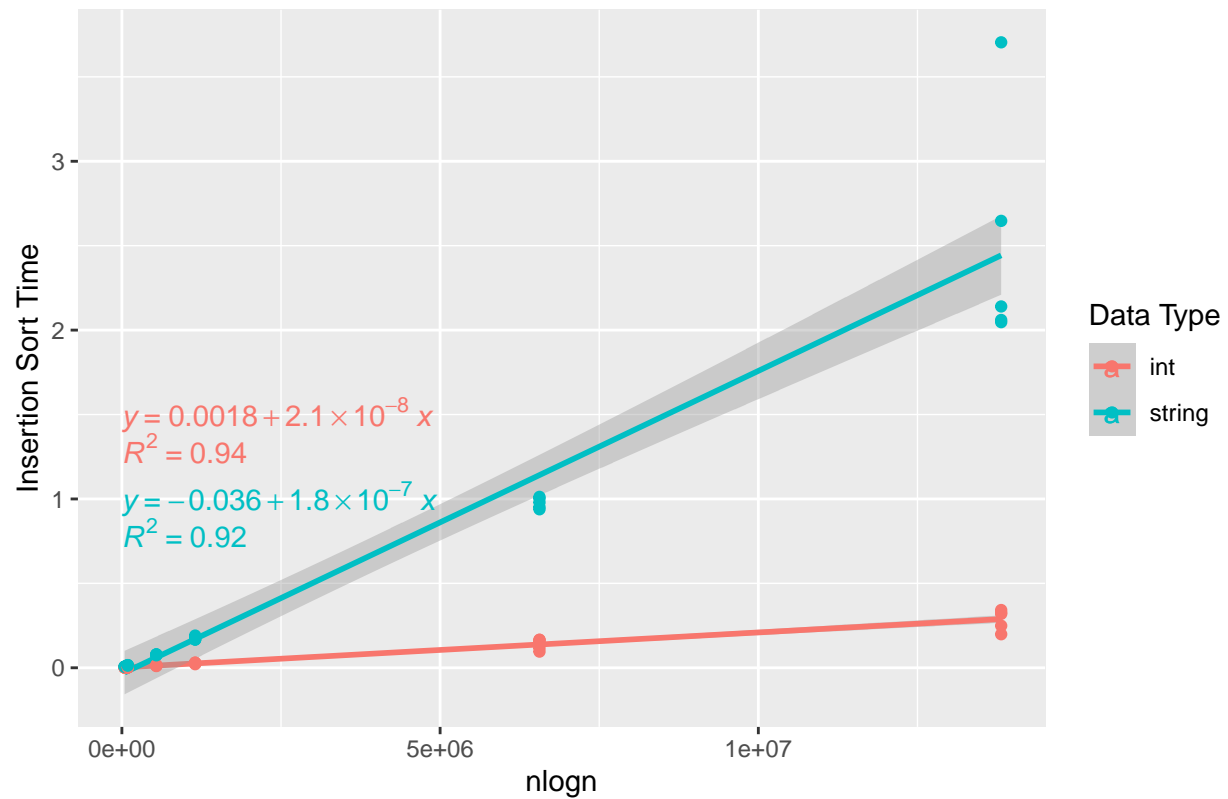
```
quickTimes = aggregate(quick_time ~ var_type + size + nlogn + format, data = data, FUN = mean)
quickTimes2 = aggregate(quick_time ~ var_type + size + nlogn, data = data, FUN = mean)
ggplot(quickTimes2, aes(x = size, y = quick_time, color = var_type)) +
  geom_line() +
  labs(title = "Mean Quick Sort Time By Data Set Size and Data Type", x = "n", y = "Quick Sort Time") +
  guides(color = guide_legend(title = "Data Type"))
```



```
ggplot(quickTimes, aes(x = nlogn, y = quick_time, color = var_type)) +
  labs(title = "Quick Sort Regression Models By Data Type", x = "nlogn", y = "Insertion Sort Time") +
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(1.5, 1)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(1.3, 0.8)) +
  guides(color = guide_legend(title = "Data Type"))
```

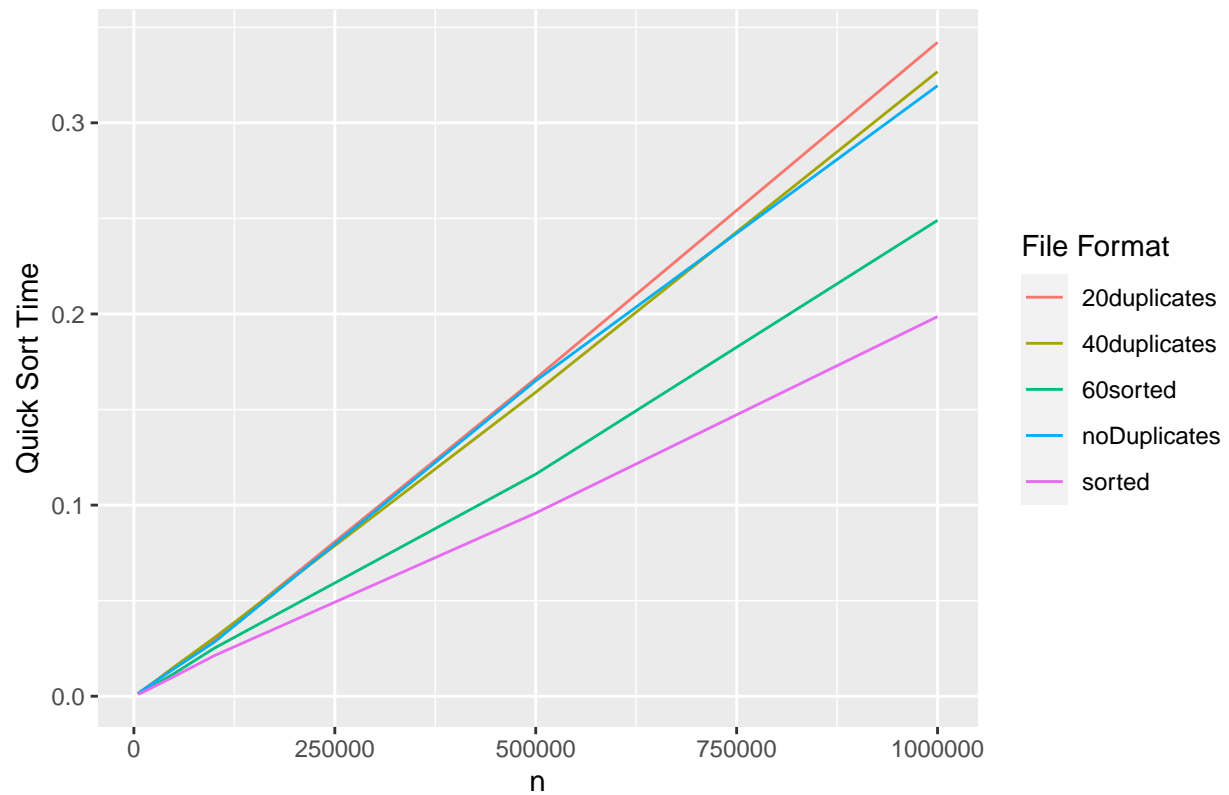
```
## 'geom_smooth()' using formula 'y ~ x'
```


Quick Sort Regression Models By Data Type



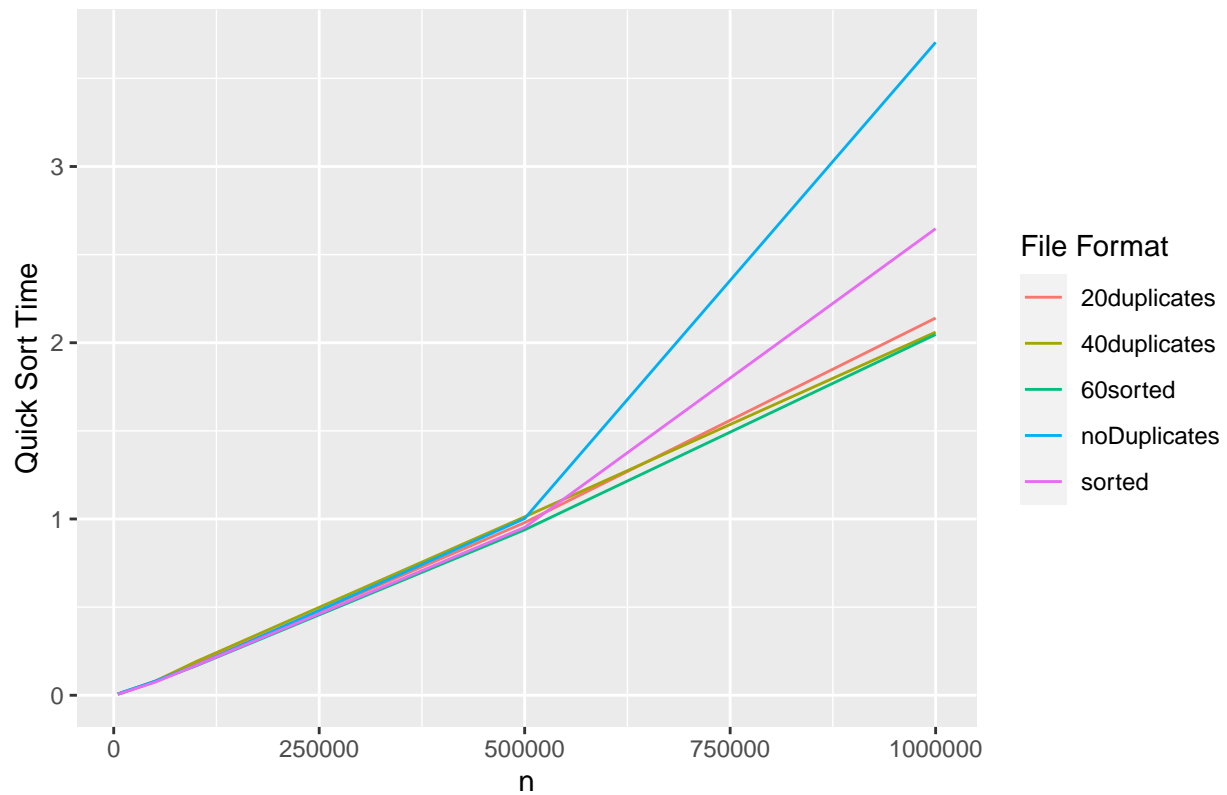
```
quickInts = subset(quickTimes, var_type == "int")
ggplot(quickInts, aes(x = size, y = quick_time, color = format)) +
  geom_line() +
  labs(title = "Quick Sort Time With Integer Data By Data Set Size and File Format", x = "n", y = "Quick")
guides(color = guide_legend(title = "File Format"))
```

Quick Sort Time With Integer Data By Data Set Size and File Format



```
quickStrings = subset(quickTimes, var_type == "string")
ggplot(quickStrings, aes(x = size, y = quick_time, color = format)) +
  geom_line() +
  labs(title = "Quick Sort Time With String Data By Data Set Size and File Format", x = "n", y = "Quick")
guides(color = guide_legend(title = "File Format"))
```

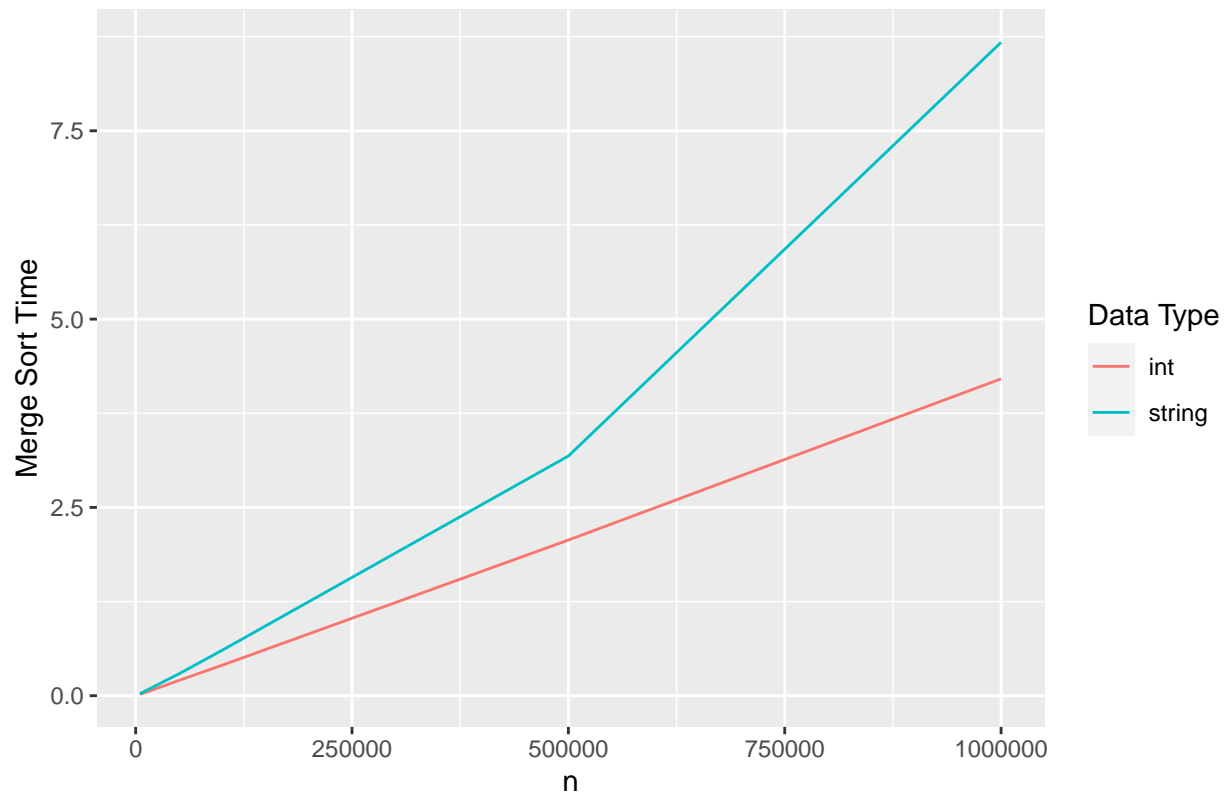
Quick Sort Time With String Data By Data Set Size and File Format



Merge Sort

```
mergeTimes = aggregate(merge_time ~ var_type + size + nlogn + format, data = data, FUN = mean)
mergeTimes2 = aggregate(merge_time ~ var_type + size + nlogn, data = data, FUN = mean)
ggplot(mergeTimes2, aes(x = size, y = merge_time, color = var_type)) +
  geom_line() +
  labs(title = "Mean Merge Sort Time By Data Set Size and Data Type", x = "n", y = "Merge Sort Time") +
  guides(color = guide_legend(title = "Data Type"))
```

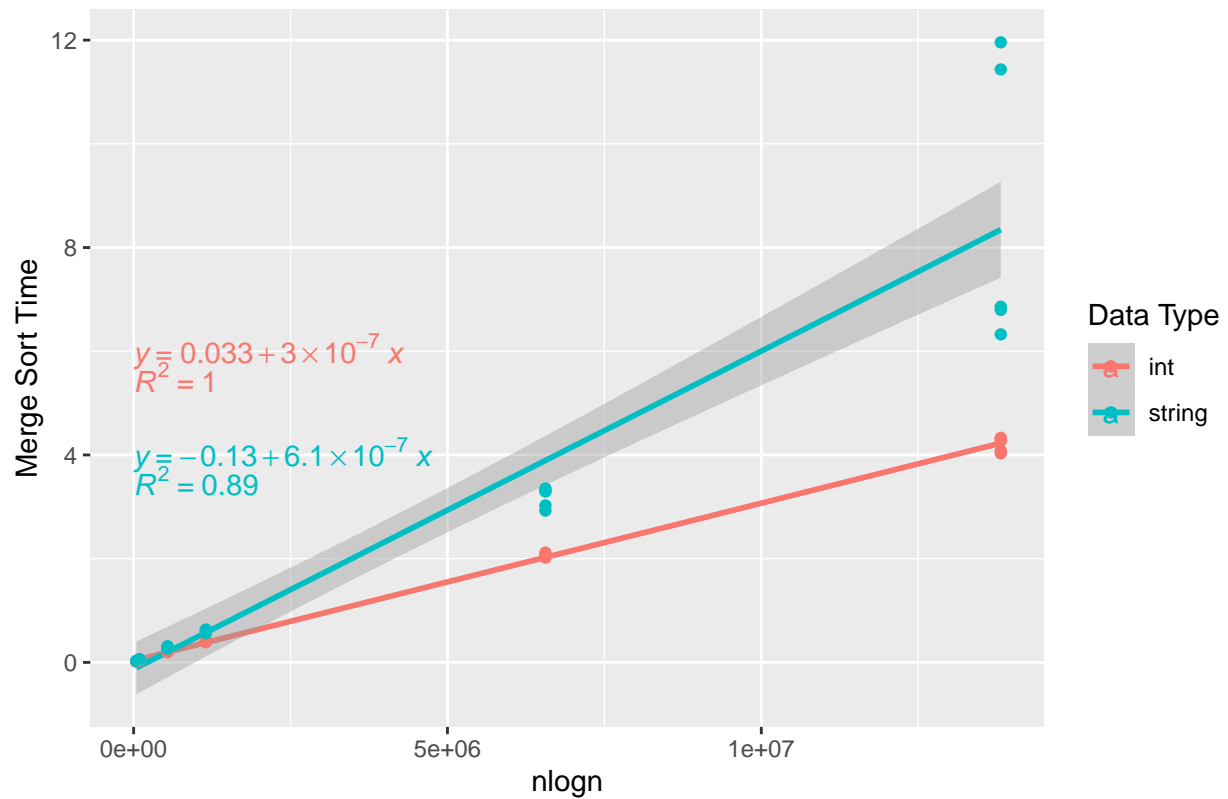
Mean Merge Sort Time By Data Set Size and Data Type



```
ggplot(mergeTimes, aes(x = nlogn, y = merge_time, color = var_type)) +  
  labs(title = "Merge Sort Regression Models By Data Type", x = "nlogn", y = "Merge Sort Time") +  
  geom_smooth(method="lm") +  
  geom_point() +  
  stat_regline_equation(label.x=0, label.y=c(6, 4)) +  
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(5.5, 3.5)) +  
  guides(color = guide_legend(title = "Data Type"))
```

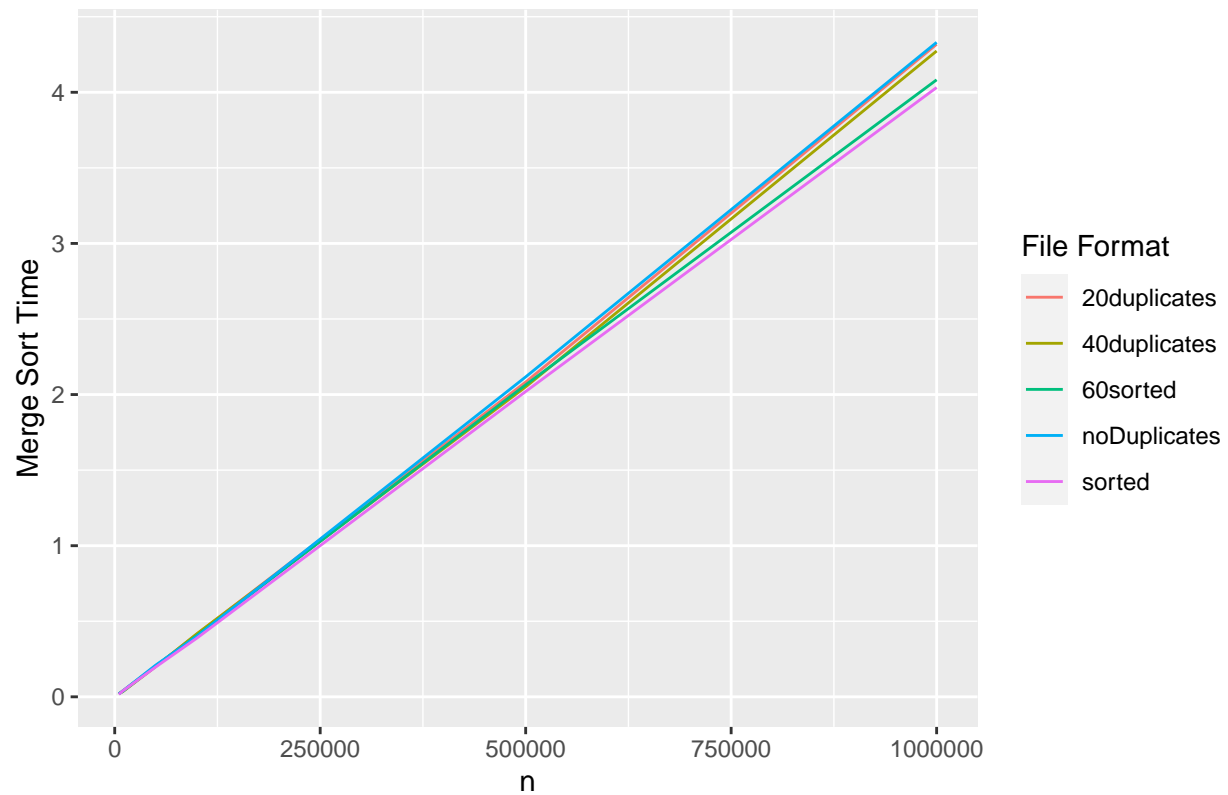
```
## 'geom_smooth()' using formula 'y ~ x'
```

Merge Sort Regression Models By Data Type

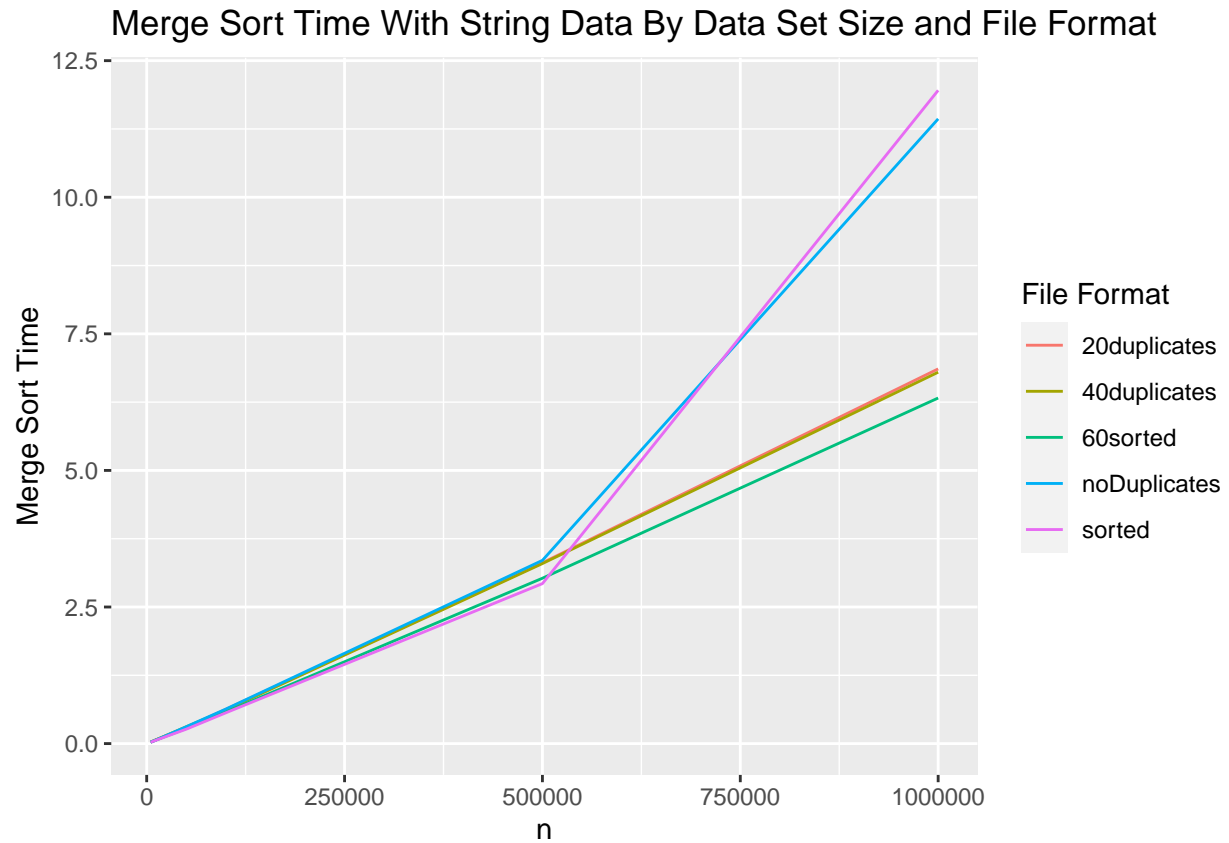


```
mergeInts = subset(mergeTimes, var_type == "int")
ggplot(mergeInts, aes(x = size, y = merge_time, color = format)) +
  geom_line() +
  labs(title = "Merge Sort Time With Integer Data By Data Set Size and File Format", x = "n", y = "Merge")
guides(color = guide_legend(title = "File Format"))
```

Merge Sort Time With Integer Data By Data Set Size and File Format

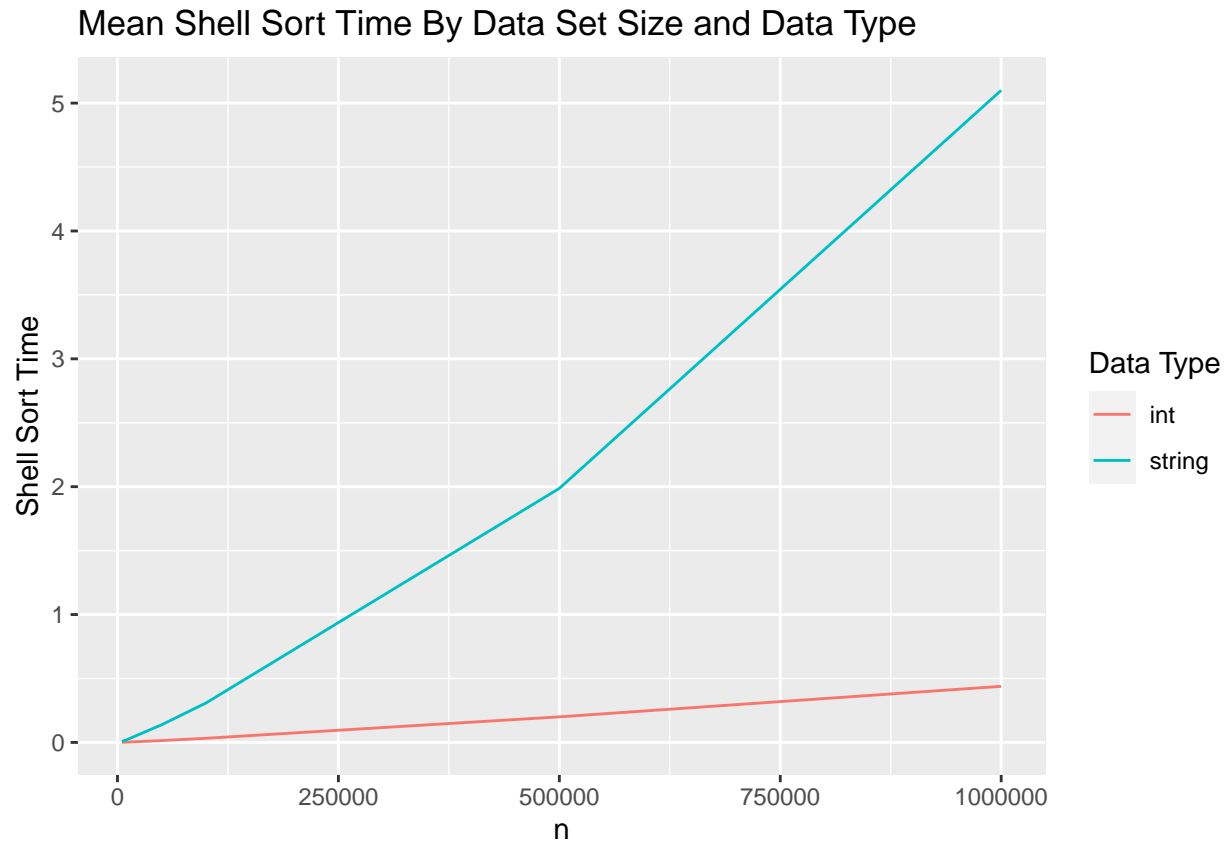


```
mergeStrings = subset(mergeTimes, var_type == "string")
ggplot(mergeStrings, aes(x = size, y = merge_time, color = format)) +
  geom_line() +
  labs(title = "Merge Sort Time With String Data By Data Set Size and File Format", x = "n", y = "Merge")
guides(color = guide_legend(title = "File Format"))
```



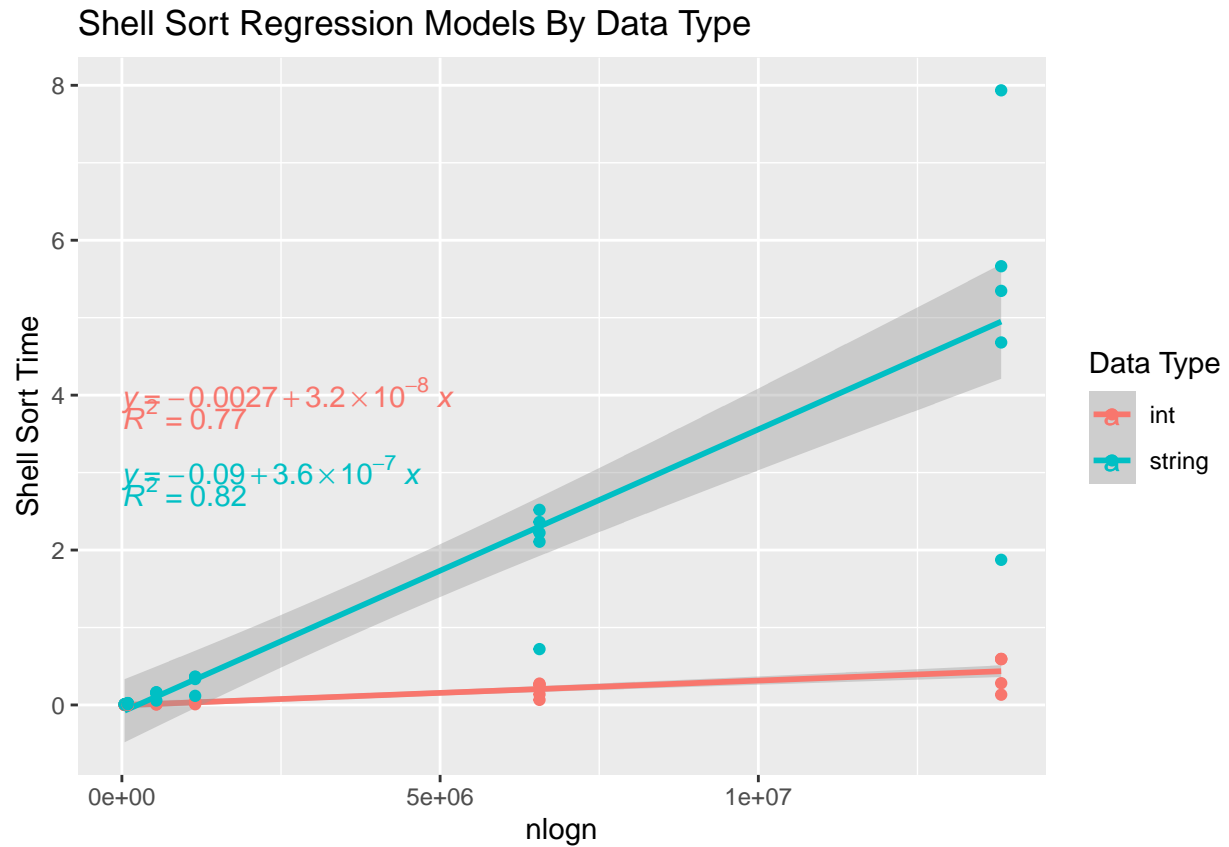
Shell Sort

```
shellTimes = aggregate(shell_time ~ var_type + size + nlogn + format, data = data, FUN = mean)
shellTimes2 = aggregate(shell_time ~ var_type + size + nlogn, data = data, FUN = mean)
ggplot(shellTimes2, aes(x = size, y = shell_time, color = var_type)) +
  geom_line() +
  labs(title = "Mean Shell Sort Time By Data Set Size and Data Type", x = "n", y = "Shell Sort Time") +
  guides(color = guide_legend(title = "Data Type"))
```



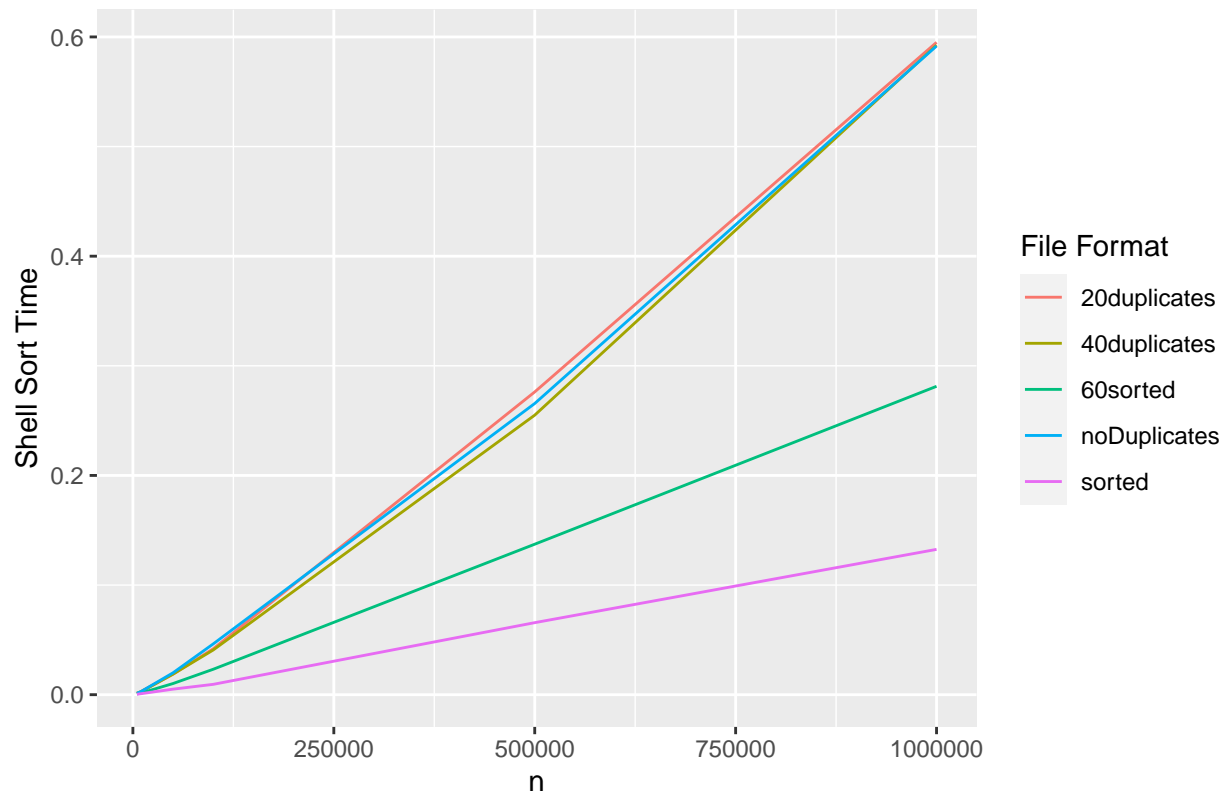
```
ggplot(shellTimes, aes(x = nlogn, y = shell_time, color = var_type)) +  
  labs(title = "Shell Sort Regression Models By Data Type", x = "nlogn", y = "Shell Sort Time") +  
  geom_smooth(method="lm") +  
  geom_point() +  
  stat_regline_equation(label.x=0, label.y=c(4, 3)) +  
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(3.75, 2.75)) +  
  guides(color = guide_legend(title = "Data Type"))
```

```
## 'geom_smooth()' using formula 'y ~ x'
```

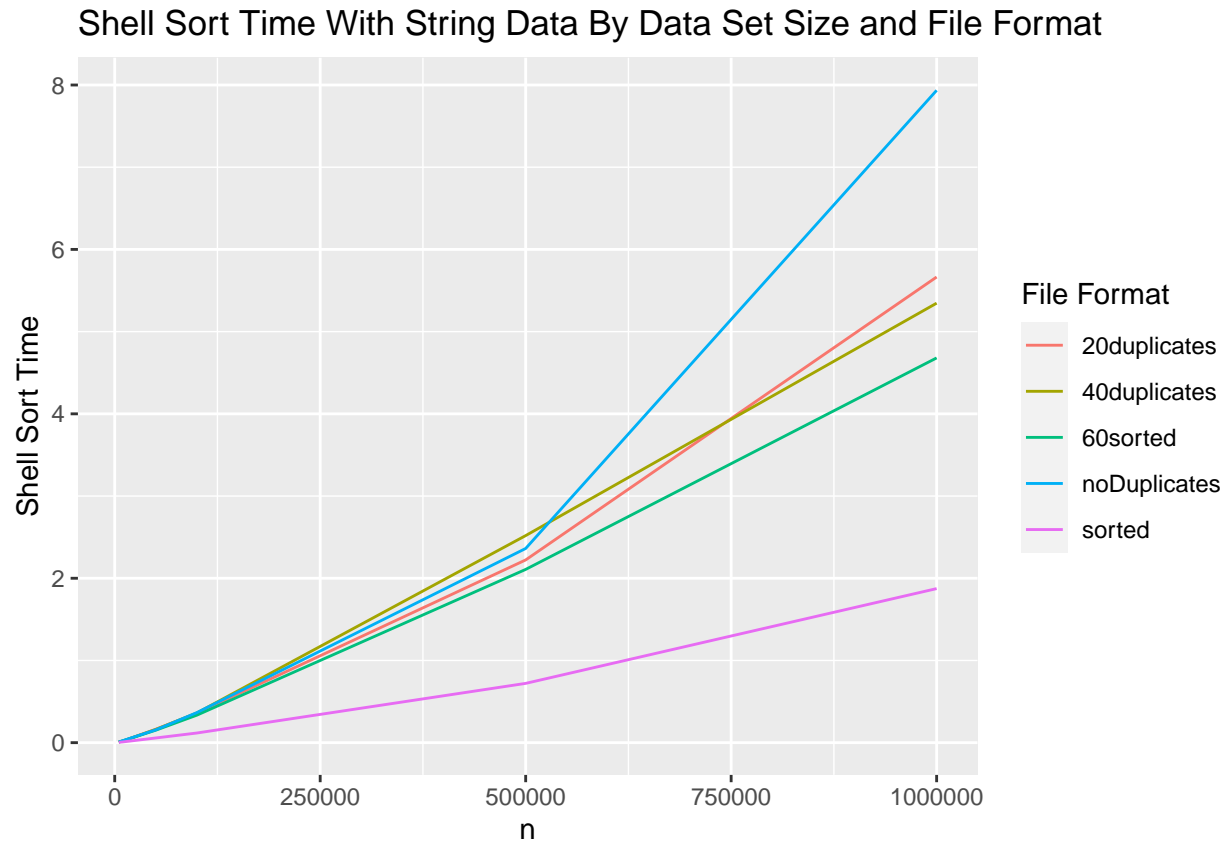



```
shellInts = subset(shellTimes, var_type == "int")
ggplot(shellInts, aes(x = size, y = shell_time, color = format)) +
  geom_line() +
  labs(title = "Shell Sort Time With Integer Data By Data Set Size and File Format", x = "n", y = "Shell Sort Time") +
  guides(color = guide_legend(title = "File Format"))
```

Shell Sort Time With Integer Data By Data Set Size and File Format

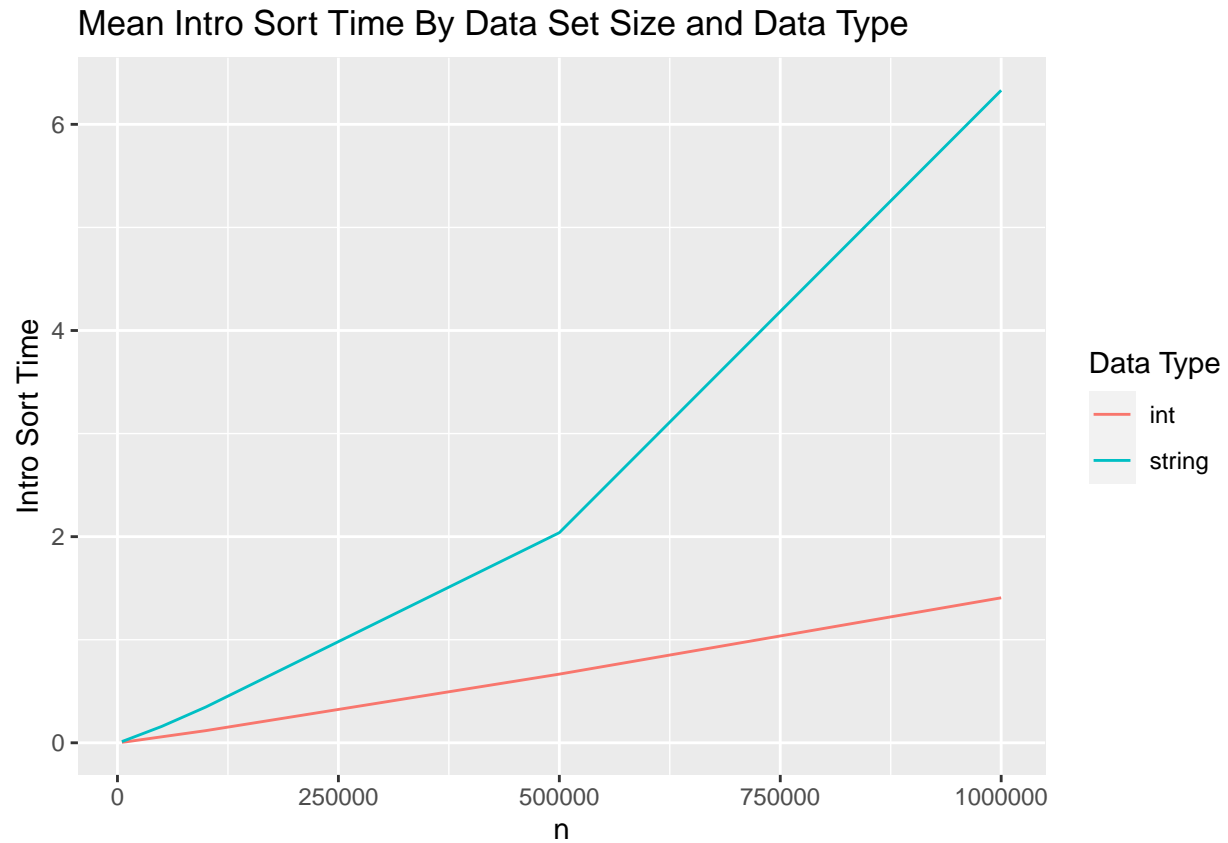


```
shellStrings = subset(shellTimes, var_type == "string")
ggplot(shellStrings, aes(x = size, y = shell_time, color = format)) +
  geom_line() +
  labs(title = "Shell Sort Time With String Data By Data Set Size and File Format", x = "n", y = "Shell")
guides(color = guide_legend(title = "File Format"))
```



Intro Sort

```
introTimes = aggregate(intro_time ~ var_type + size + nlogn + format, data = data, FUN = mean)
introTimes2 = aggregate(intro_time ~ var_type + size + nlogn, data = data, FUN = mean)
ggplot(introTimes2, aes(x = size, y = intro_time, color = var_type)) +
  geom_line() +
  labs(title = "Mean Intro Sort Time By Data Set Size and Data Type", x = "n", y = "Intro Sort Time") +
  guides(color = guide_legend(title = "Data Type"))
```

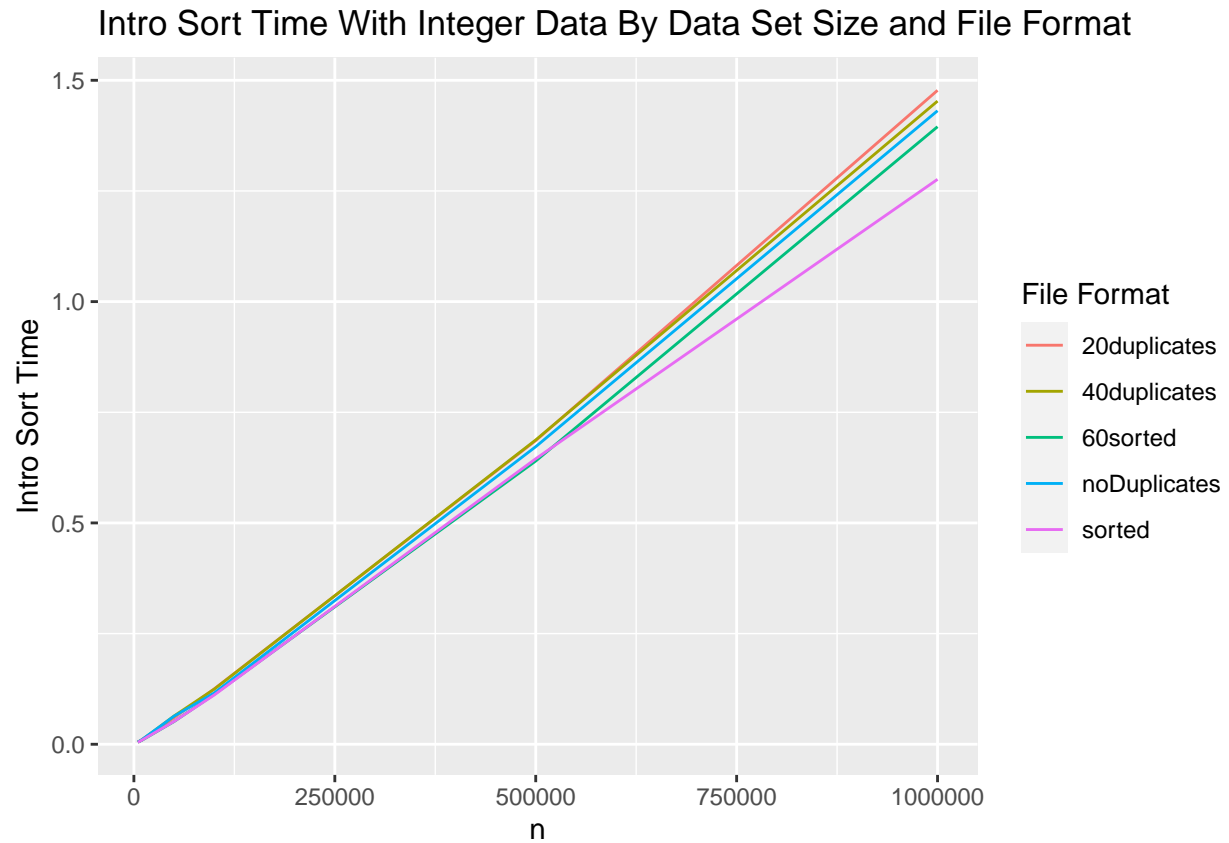


```
ggplot(introTimes, aes(x = nlogn, y = intro_time, color = var_type)) +  
  labs(title = "Intro Sort Regression Models By Data Type", x = "nlogn", y = "Intro Sort Time") +  
  geom_smooth(method="lm") +  
  geom_point() +  
  stat_regline_equation(label.x=0, label.y=c(4, 3)) +  
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(3.75, 2.75)) +  
  guides(color = guide_legend(title = "Data Type"))
```

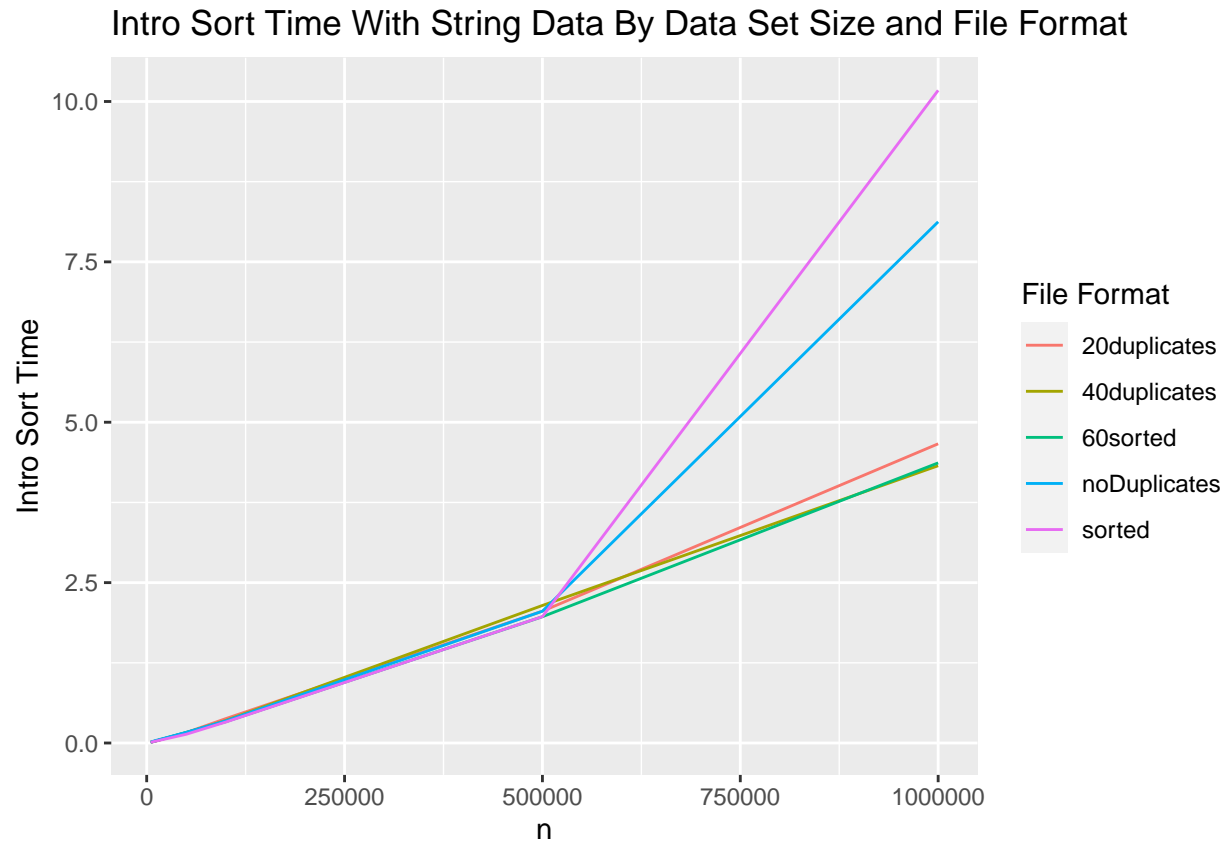
```
## 'geom_smooth()' using formula 'y ~ x'
```



```
introInts = subset(introTimes, var_type == "int")
ggplot(introInts, aes(x = size, y = intro_time, color = format)) +
  geom_line() +
  labs(title = "Intro Sort Time With Integer Data By Data Set Size and File Format", x = "n", y = "Intro")
guides(color = guide_legend(title = "File Format"))
```



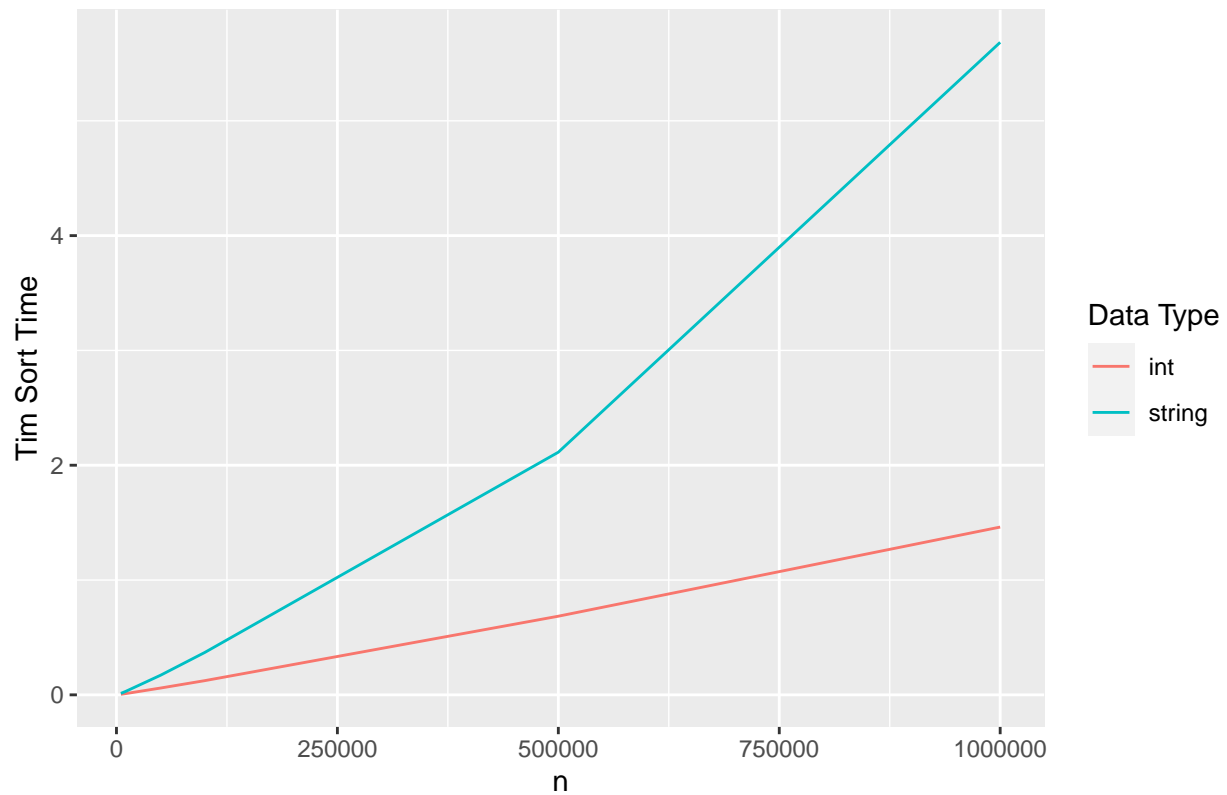
```
introStrings = subset(introTimes, var_type == "string")
ggplot(introStrings, aes(x = size, y = intro_time, color = format)) +
  geom_line() +
  labs(title = "Intro Sort Time With String Data By Data Set Size and File Format", x = "n", y = "Intro")
guides(color = guide_legend(title = "File Format"))
```



Tim Sort

```
timTimes = aggregate(tim_time ~ var_type + size + nlogn + format, data = data, FUN = mean)
timTimes2 = aggregate(tim_time ~ var_type + size + nlogn, data = data, FUN = mean)
ggplot(timTimes2, aes(x = size, y = tim_time, color = var_type)) +
  geom_line() +
  labs(title = "Mean Tim Sort Time By Data Set Size and Data Type", x = "n", y = "Tim Sort Time") +
  guides(color = guide_legend(title = "Data Type"))
```

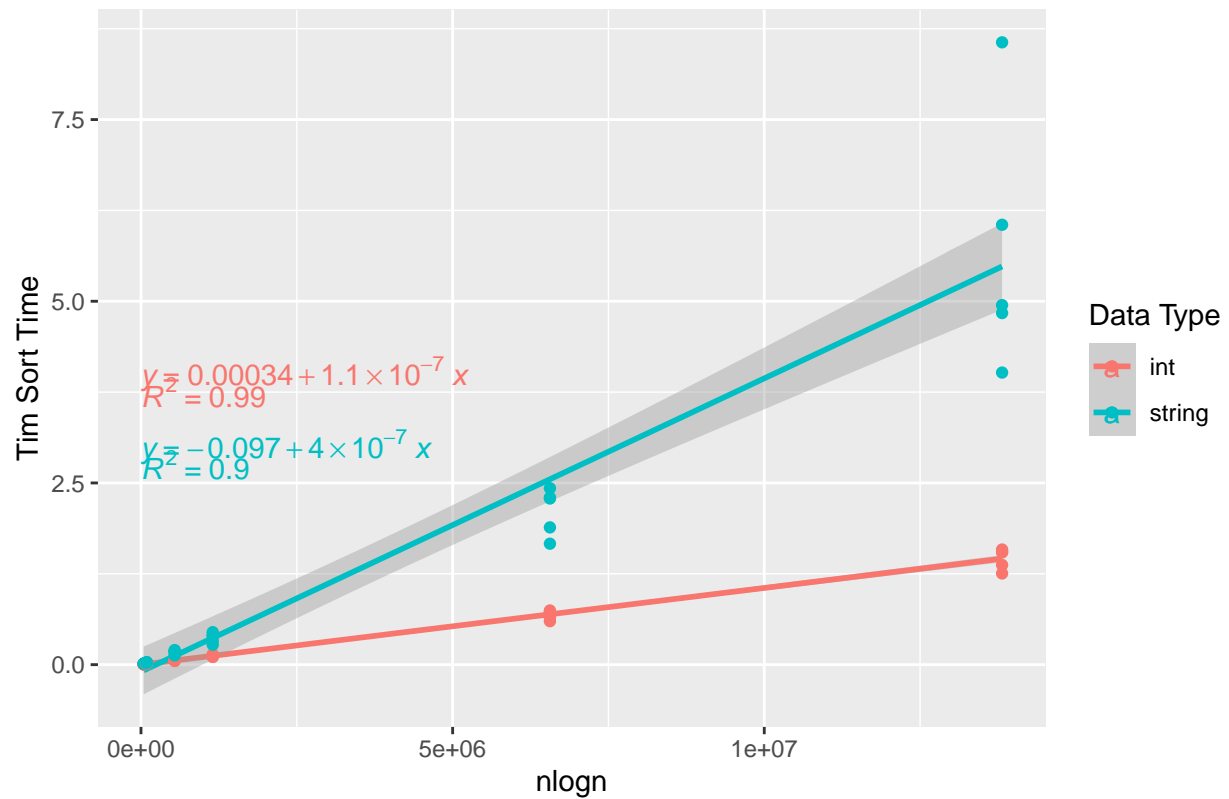
Mean Tim Sort Time By Data Set Size and Data Type



```
ggplot(timTimes, aes(x = nlogn, y = tim_time, color = var_type)) +  
  labs(title = "Tim Sort Regression Models By Data Type", x = "nlogn", y = "Tim Sort Time") +  
  geom_smooth(method="lm") +  
  geom_point() +  
  stat_regline_equation(label.x=0, label.y=c(4, 3)) +  
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(3.75, 2.75)) +  
  guides(color = guide_legend(title = "Data Type"))
```

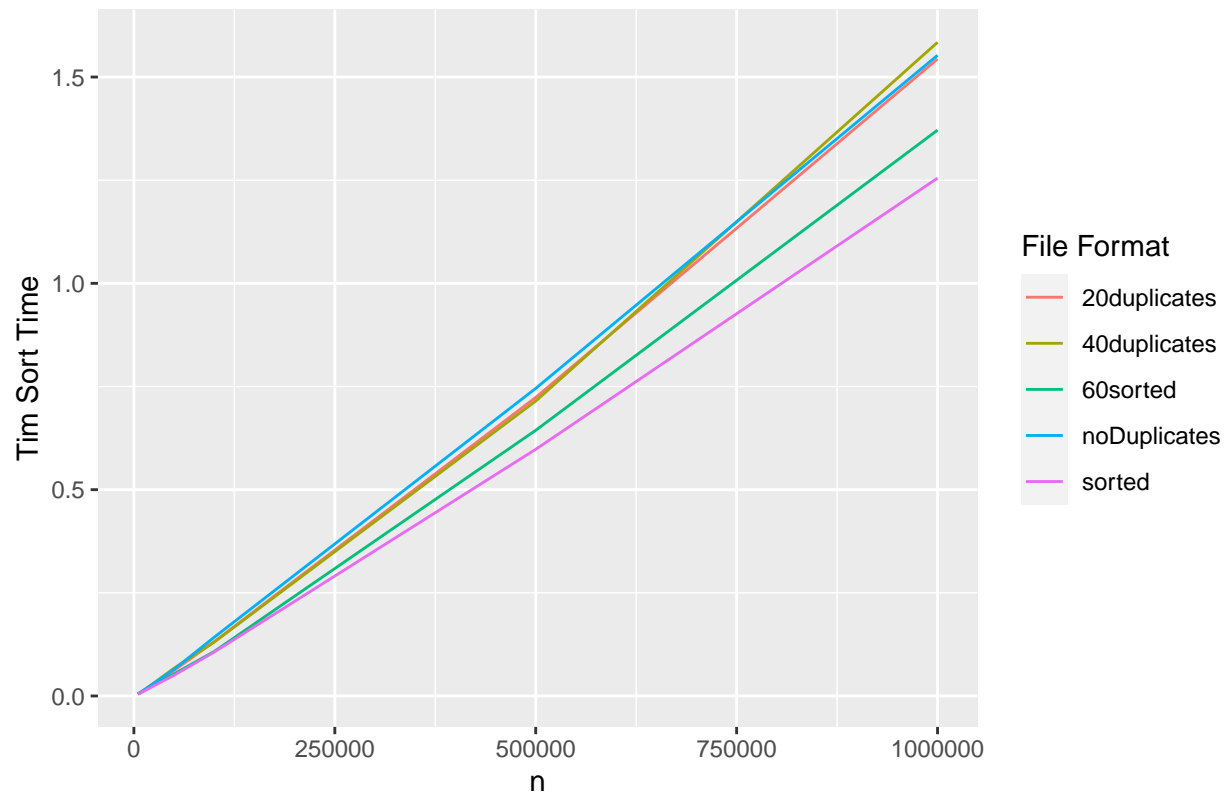
```
## 'geom_smooth()' using formula 'y ~ x'
```


Tim Sort Regression Models By Data Type



```
timInts = subset(timTimes, var_type == "int")
ggplot(timInts, aes(x = size, y = tim_time, color = format)) +
  geom_line() +
  labs(title = "Tim Sort Time With Integer Data By Data Set Size and File Format", x = "n", y = "Tim Sort Time") +
  guides(color = guide_legend(title = "File Format"))
```

Tim Sort Time With Integer Data By Data Set Size and File Format



```
timStrings = subset(timTimes, var_type == "string")
ggplot(timStrings, aes(x = size, y = tim_time, color = format)) +
  geom_line() +
  labs(title = "Tim Sort Time With String Data By Data Set Size and File Format", x = "n", y = "Tim Sort Time") +
  guides(color = guide_legend(title = "File Format"))
```

Tim Sort Time With String Data By Data Set Size and File Format

