# Program 2 Graph Analysis

Ryan Schaefer and Wes Anderson

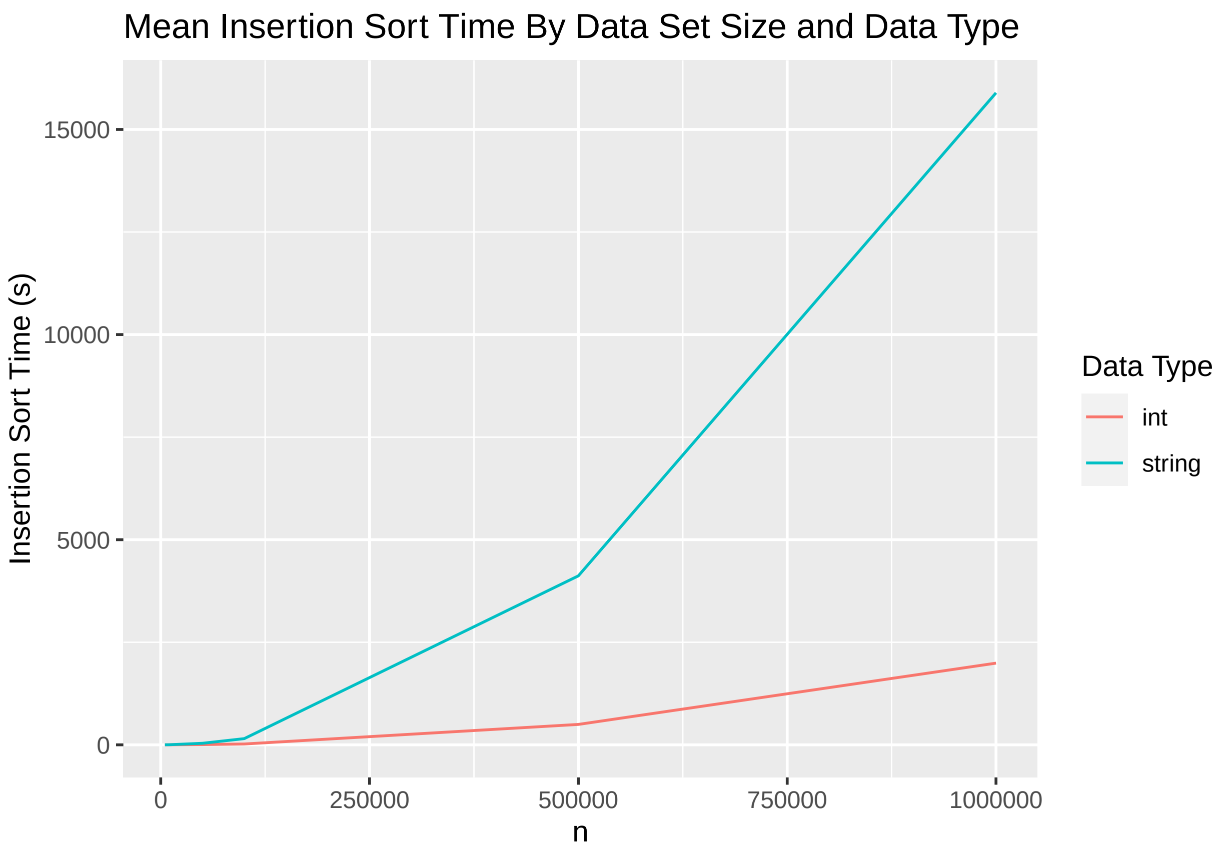
## Create Dataset

```
# Load Dependencies
library(ggplot2)
library(ggpubr)
library(svglite)

# Load Dataset
data = read.csv("FinalData.csv")
# Create n^2 and nlogn columns
data$n2 = data$size ^ 2
data$nlogn = log(data$size) * data$size
```

## 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 
  guides(color = guide_legend(title = "Data Type"))
```

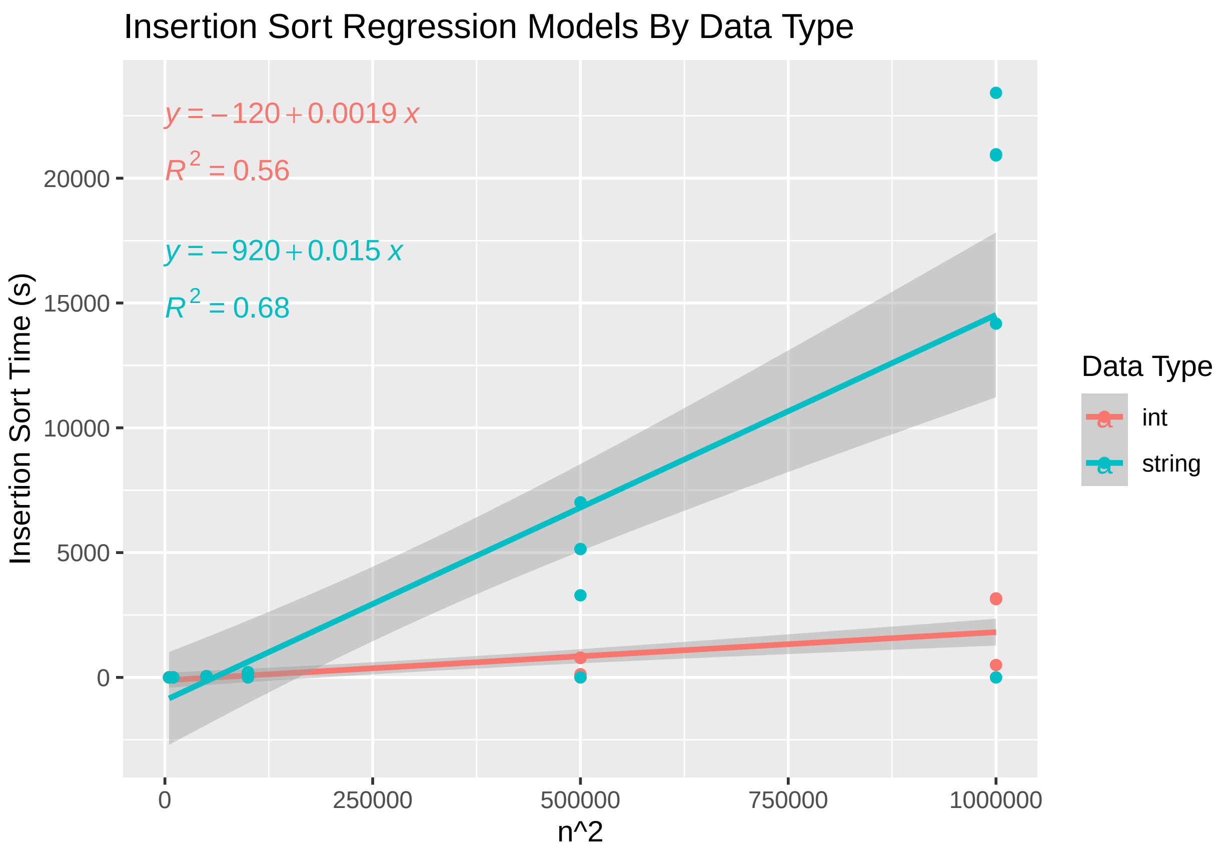


```
ggsave("insertionMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 (s)
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(22500, 17000)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(20500, 15000)) +
  guides(color = guide_legend(title = "Data Type"))
```

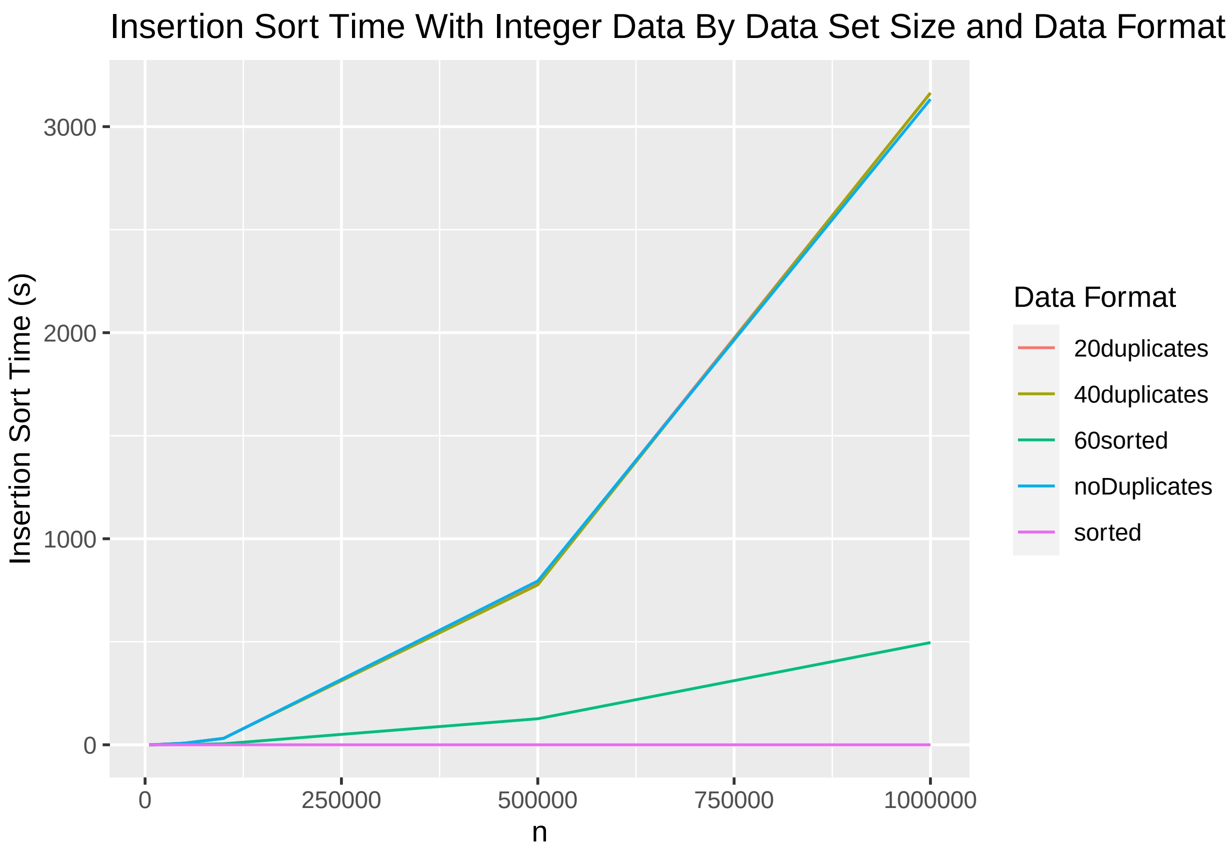
```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



```
ggsave("insertionRegression.svg")
```

```
## Saving 6.5 x 4.5 in image
## ‘geom_smooth()‘ using formula ’y ~ x’
```

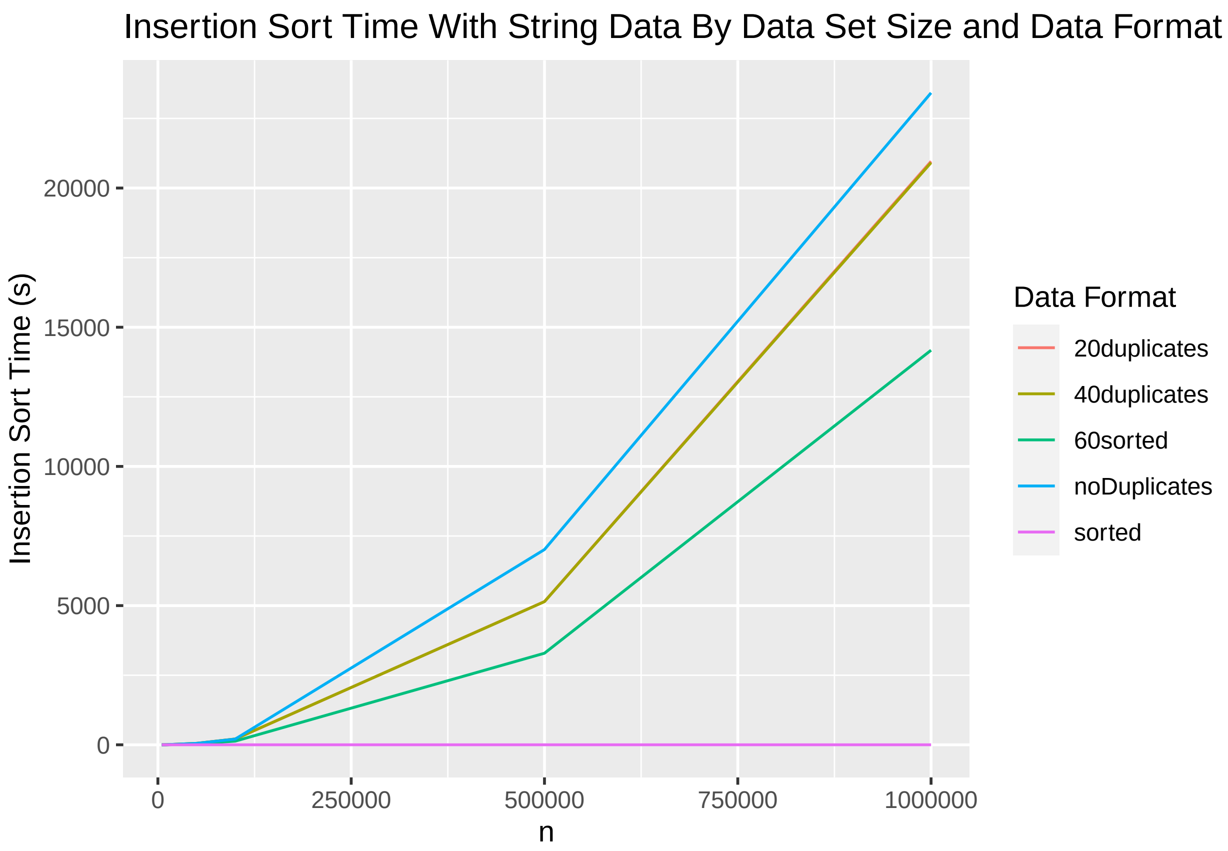
```
insertionInts = subset(insertionTimes, var_type == "int")
ggplot(insertionInts, aes(x = size, y = insertion_time, color = format)) +
  geom_line() +
  labs(title = "Insertion Sort Time With Integer Data By Data Set Size and Data Format", x = "n", y = "
  guides(color = guide_legend(title = "Data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



```
ggsave("insertionInts.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 Data Format", x = "n", y = "Ir
  guides(color = guide_legend(title = "Data Format"))
```

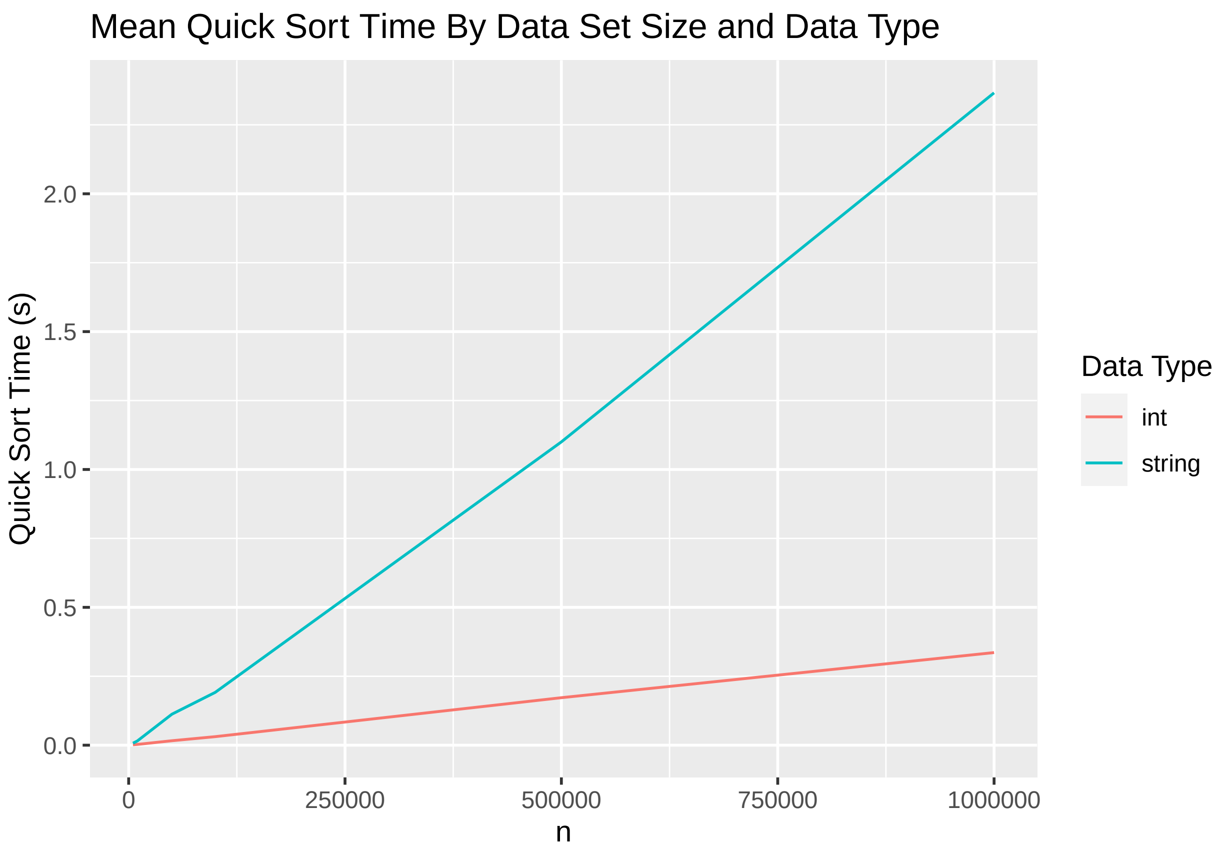


```
ggsave("insertionStrings.svg")
```

```
## Saving 6.5 x 4.5 in image
```

### 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 (s)
  guides(color = guide_legend(title = "Data Type"))
```

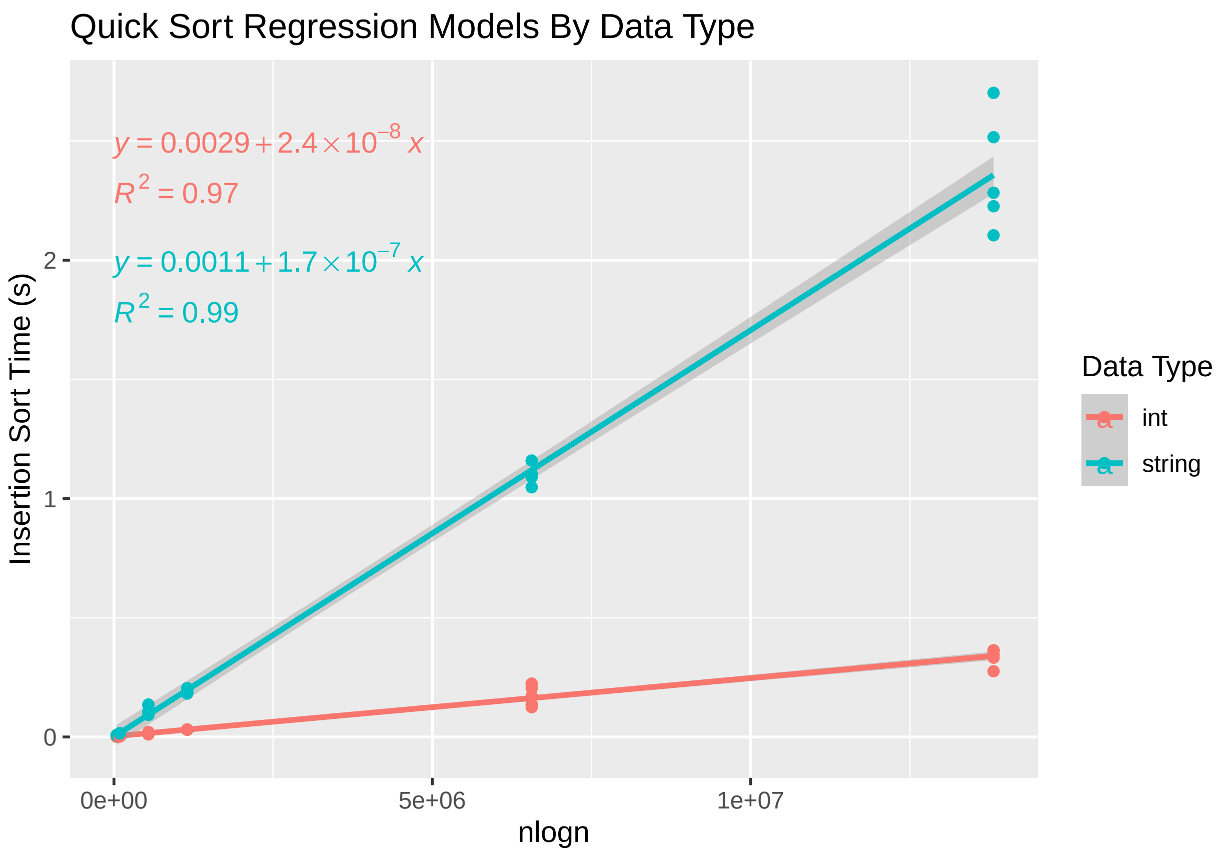


```
ggsave("quickMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 (s)")
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(2.5, 2)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(2.3, 1.8)) +
  guides(color = guide_legend(title = "Data Type"))
```

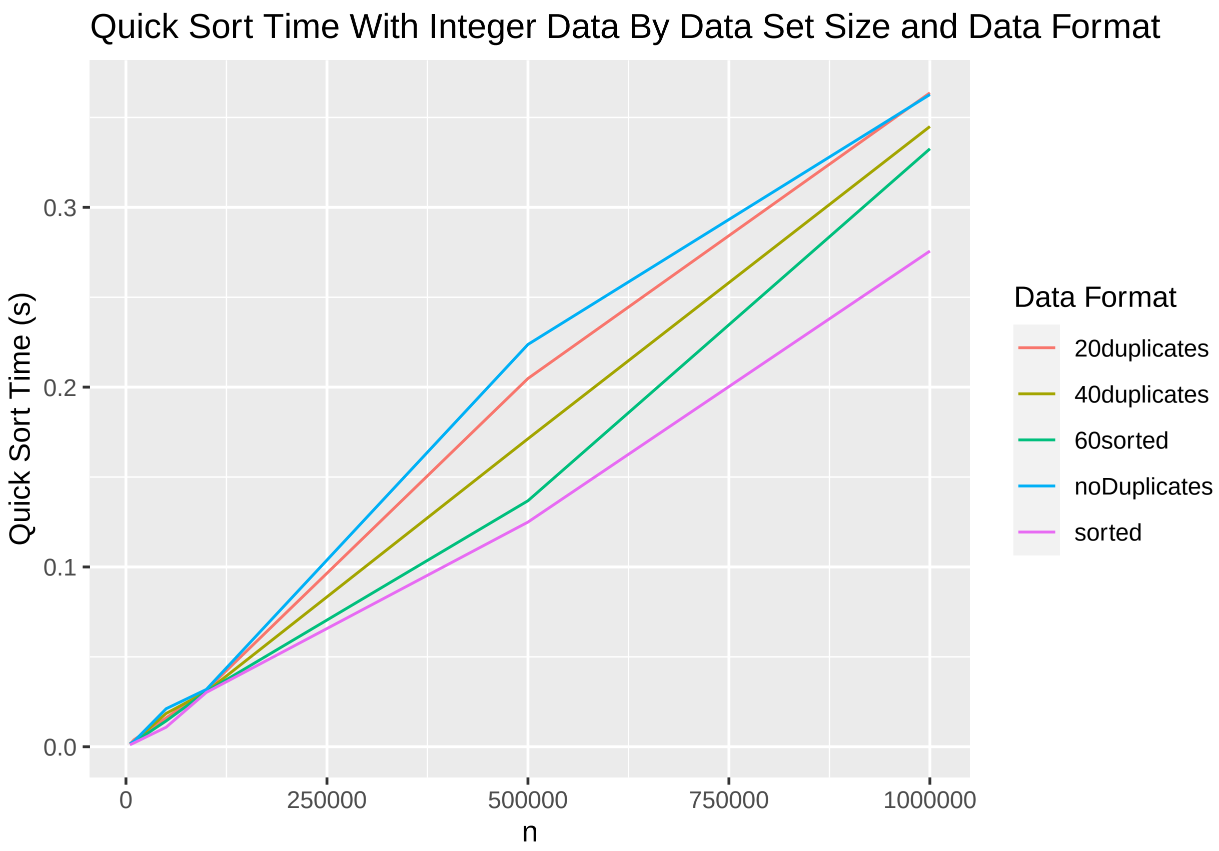
```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



```r
ggsave("quickRegression.svg")
```

```
## Saving 6.5 x 4.5 in image
## ‘geom_smooth()‘ using formula ’y ~ x’
```

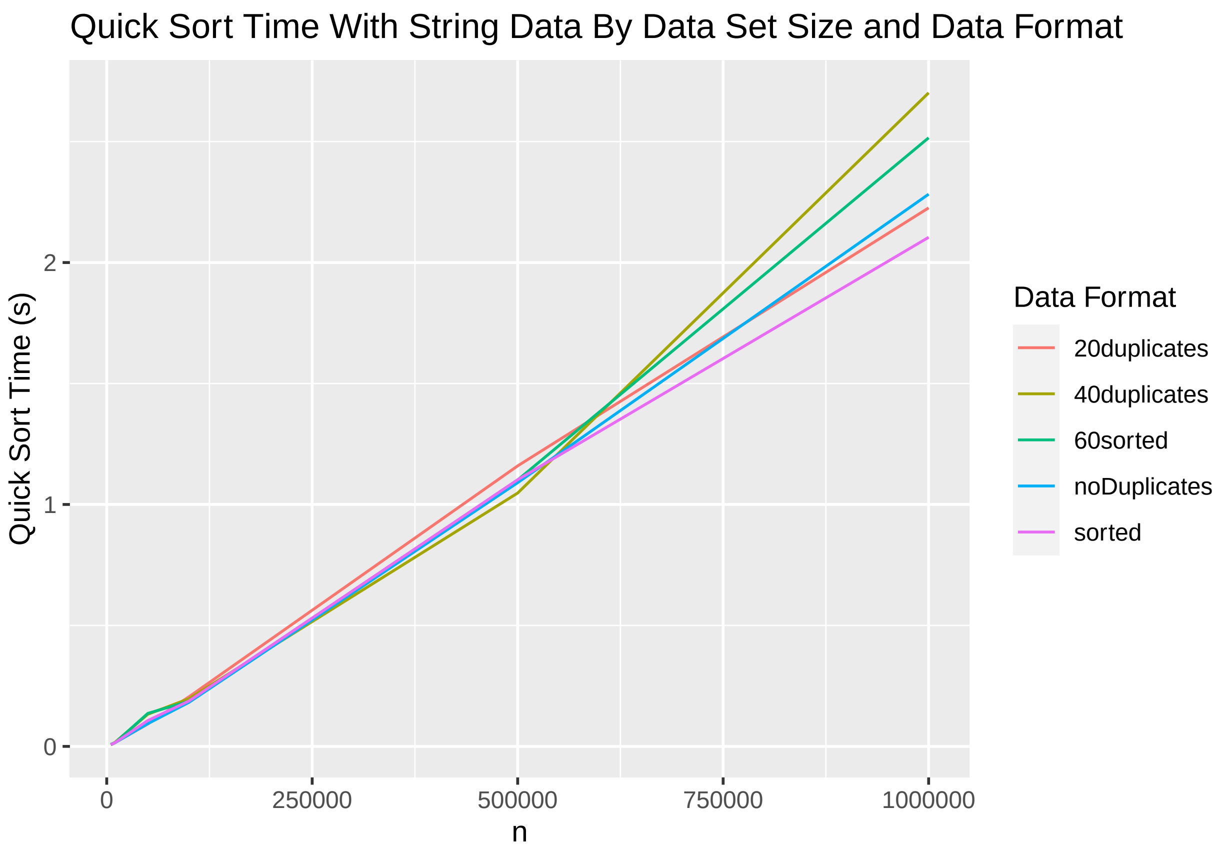
```r
quickInts = subset(quickTimes, var_type == "int")
ggplot(quickInts, aes(x = size, y = quick_time, color = format)) +
  geom_line() +
  labs(title = "Quick Sort Time With Integer Data By Data Set Size and Data Format", x = "n", y = "Quic
  guides(color = guide_legend(title = "Data Format"))
```



```
ggsave("quickInts.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 Data Format", x = "n", y = "Quick
  guides(color = guide_legend(title = "Data Format"))
```

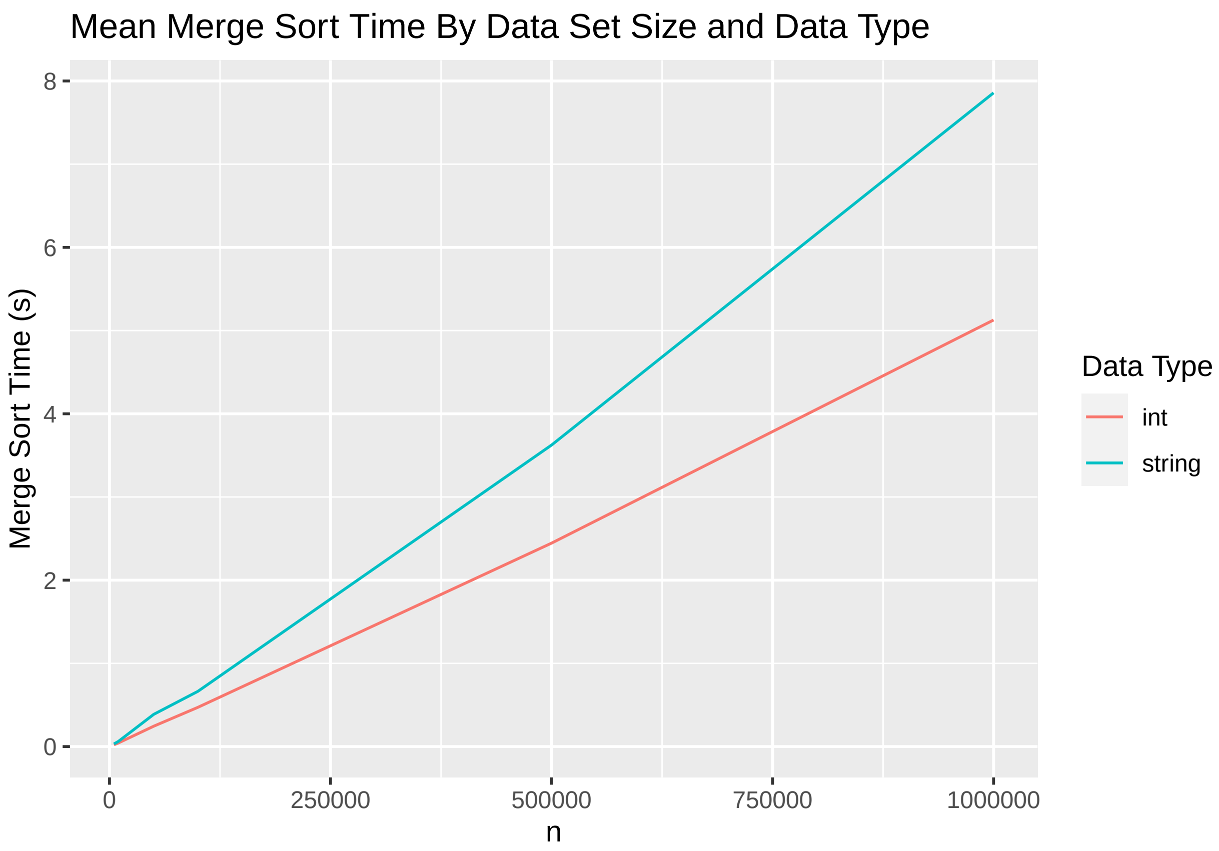


```r
ggsave("quickStrings.svg")
```

```
## Saving 6.5 x 4.5 in image
```

### Merge Sort

```r
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 (s)
  guides(color = guide_legend(title = "Data Type"))
```

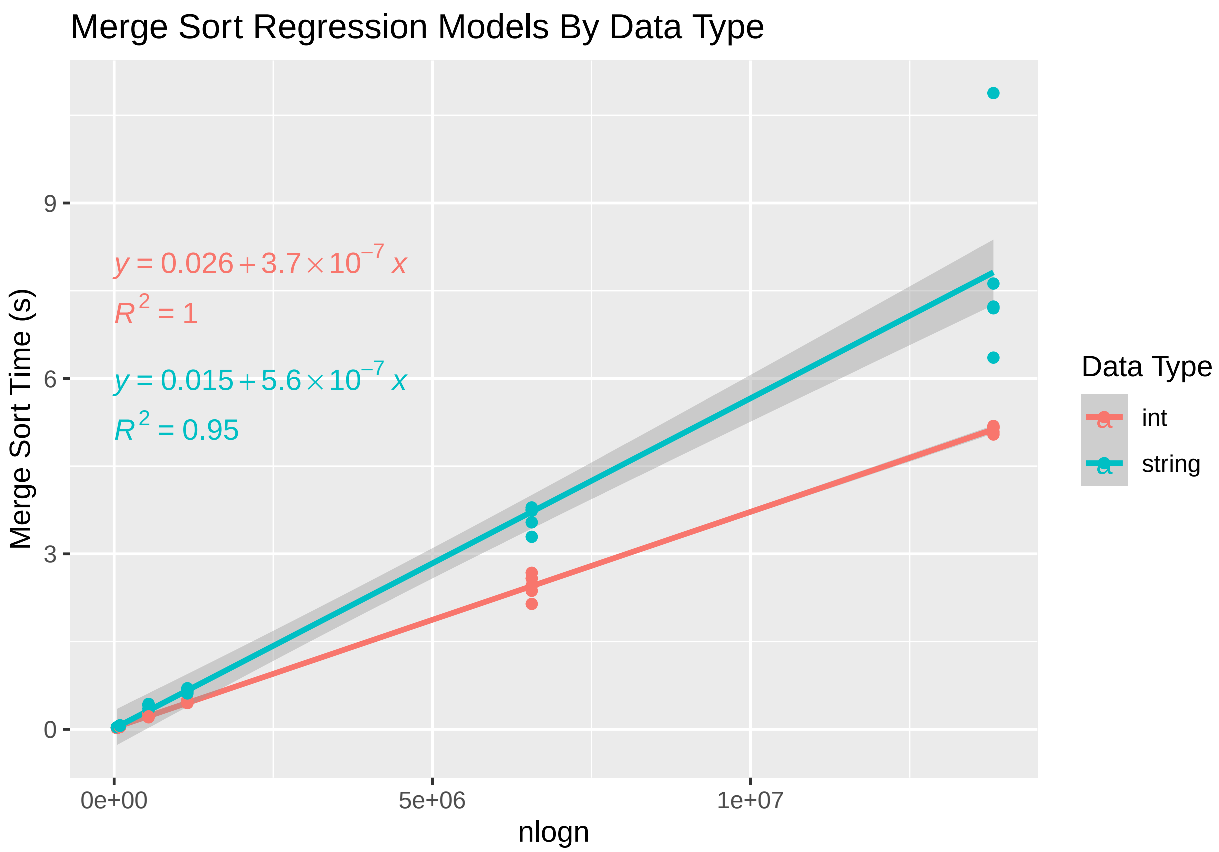


```r
ggsave("mergeMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```r
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 (s)") +
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(8, 6)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(7.2, 5.2)) +
  guides(color = guide_legend(title = "Data Type"))
```

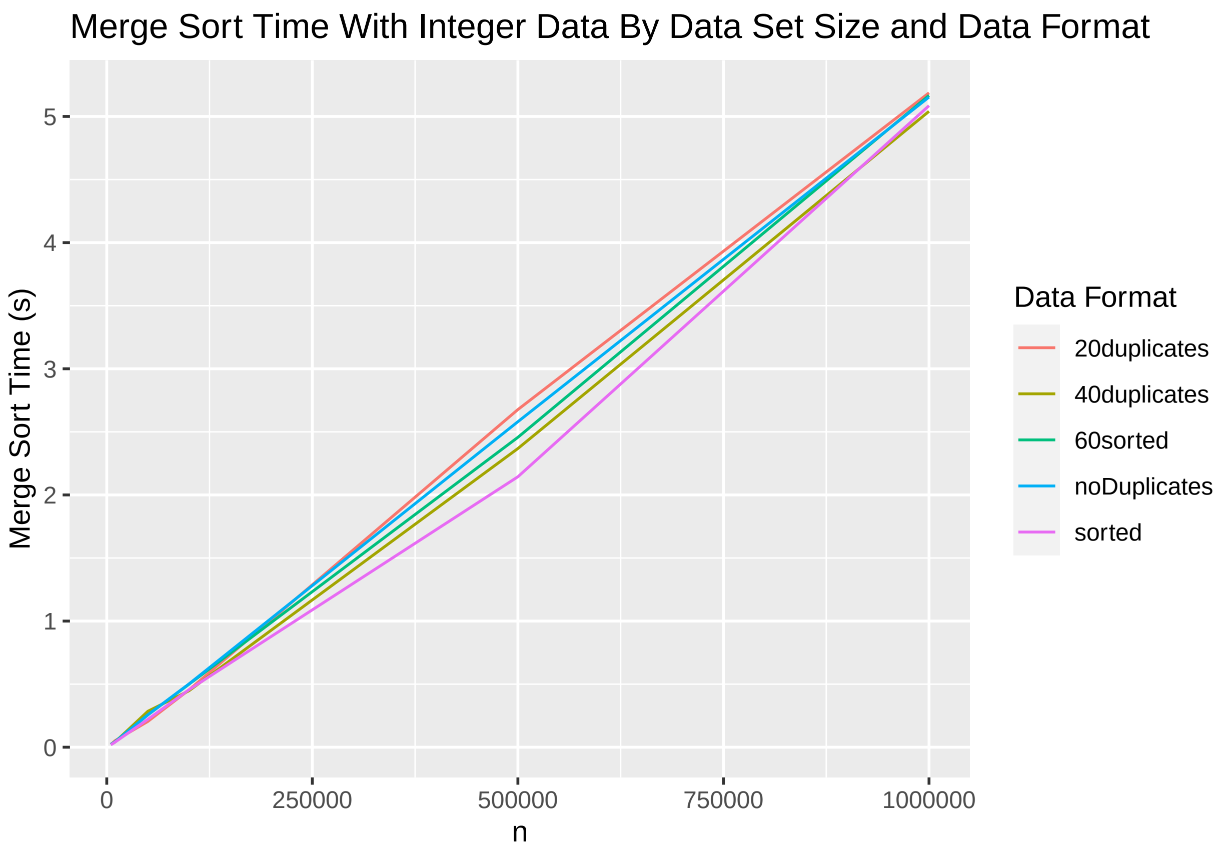
```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



```
ggsave("mergeRegression.svg")
```

```
## Saving 6.5 x 4.5 in image
## ‘geom_smooth()‘ using formula ’y ~ x’
```

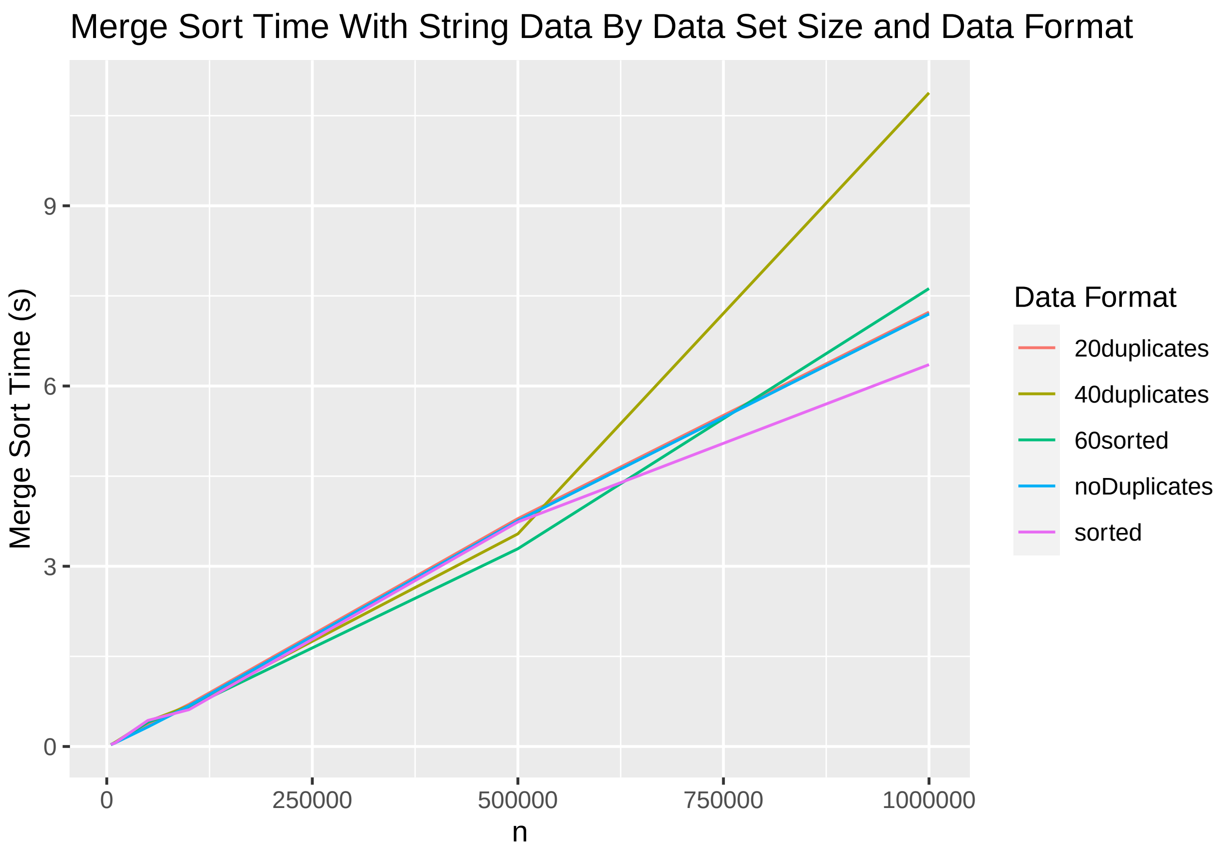
```
mergeInts = subset(mergeTimes, var_type == "int")
ggplot(mergeInts, aes(x = size, y = merge_time, color = format)) +
  geom_line() +
  labs(title = "Merge Sort Time With Integer Data By Data Set Size and Data Format", x = "n", y = "Merg
  guides(color = guide_legend(title = "Data Format"))
```



```
ggsave("mergeInts.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 Data Format", x = "n", y = "Merge
  guides(color = guide_legend(title = "Data Format"))
```

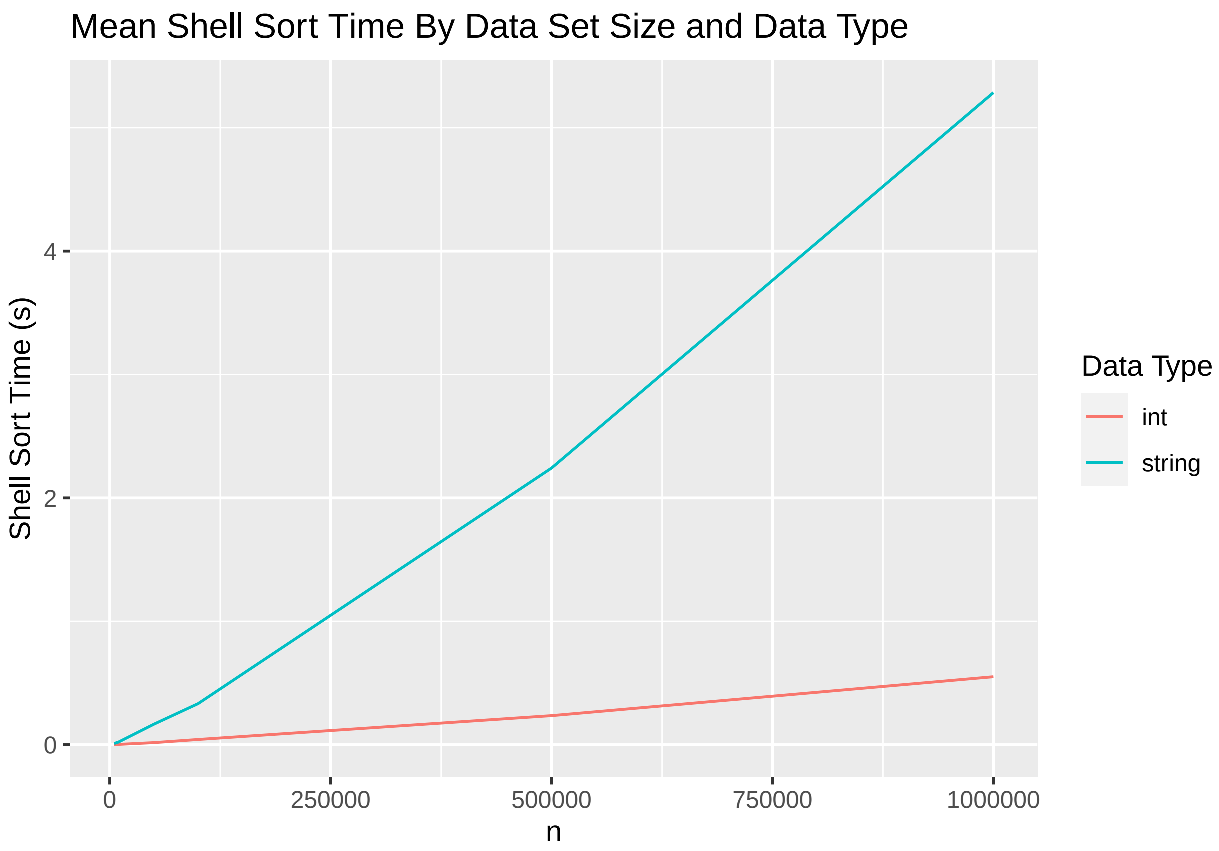


```
ggsave("mergeStrings.svg")
```

```
## Saving 6.5 x 4.5 in image
```

### 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 (s)
  guides(color = guide_legend(title = "Data Type"))
```

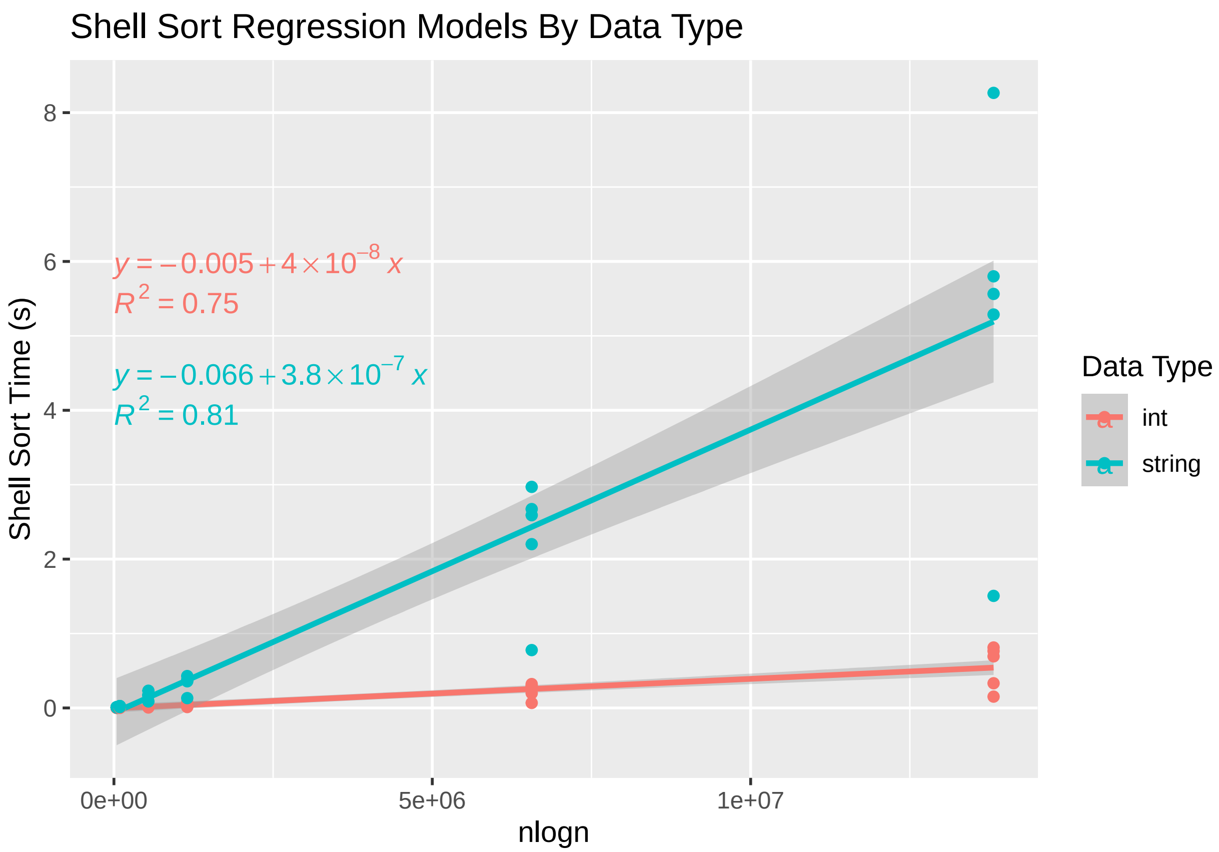


```
ggsave("shellMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 (s)") +
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(6, 4.5)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(5.5, 4)) +
  guides(color = guide_legend(title = "Data Type"))
```

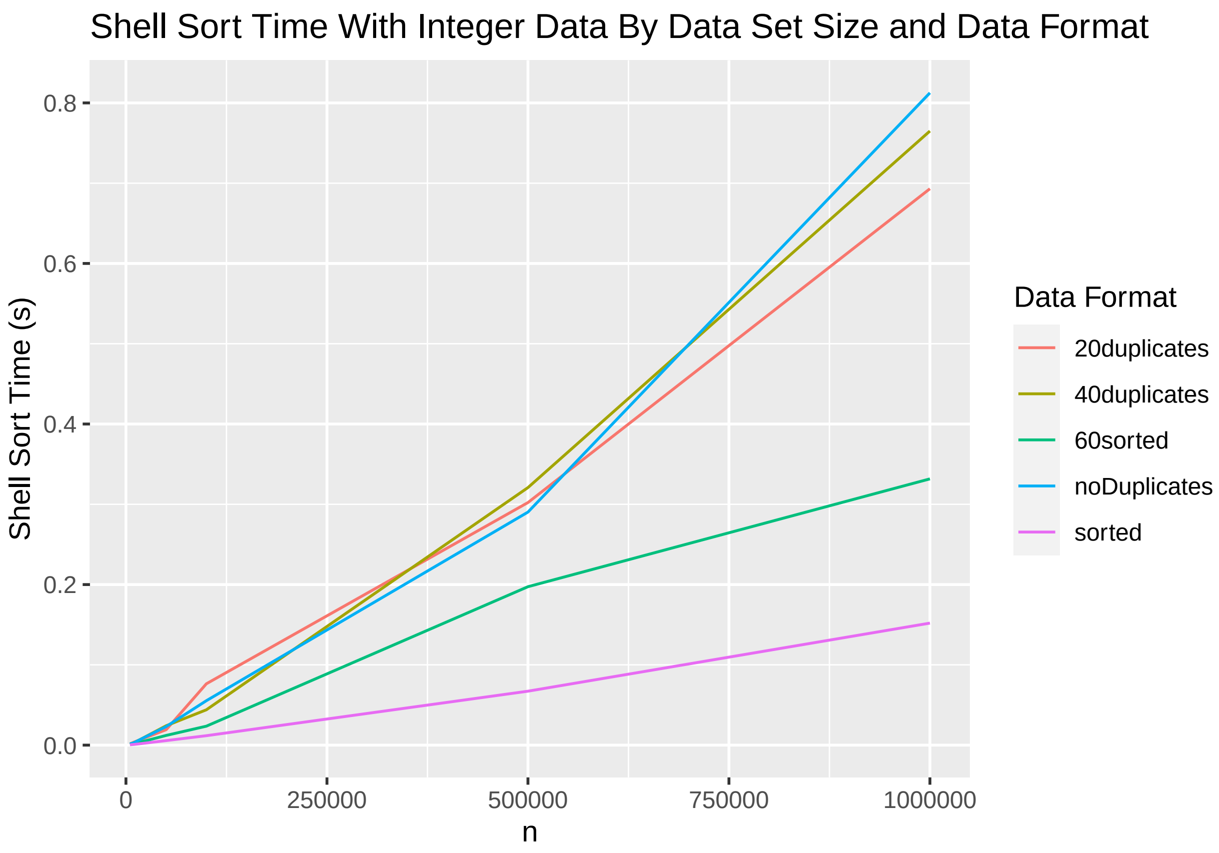
```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



```
ggsave("shellRegression.svg")
```

```
## Saving 6.5 x 4.5 in image
## ‘geom_smooth()‘ using formula ’y ~ x’
```

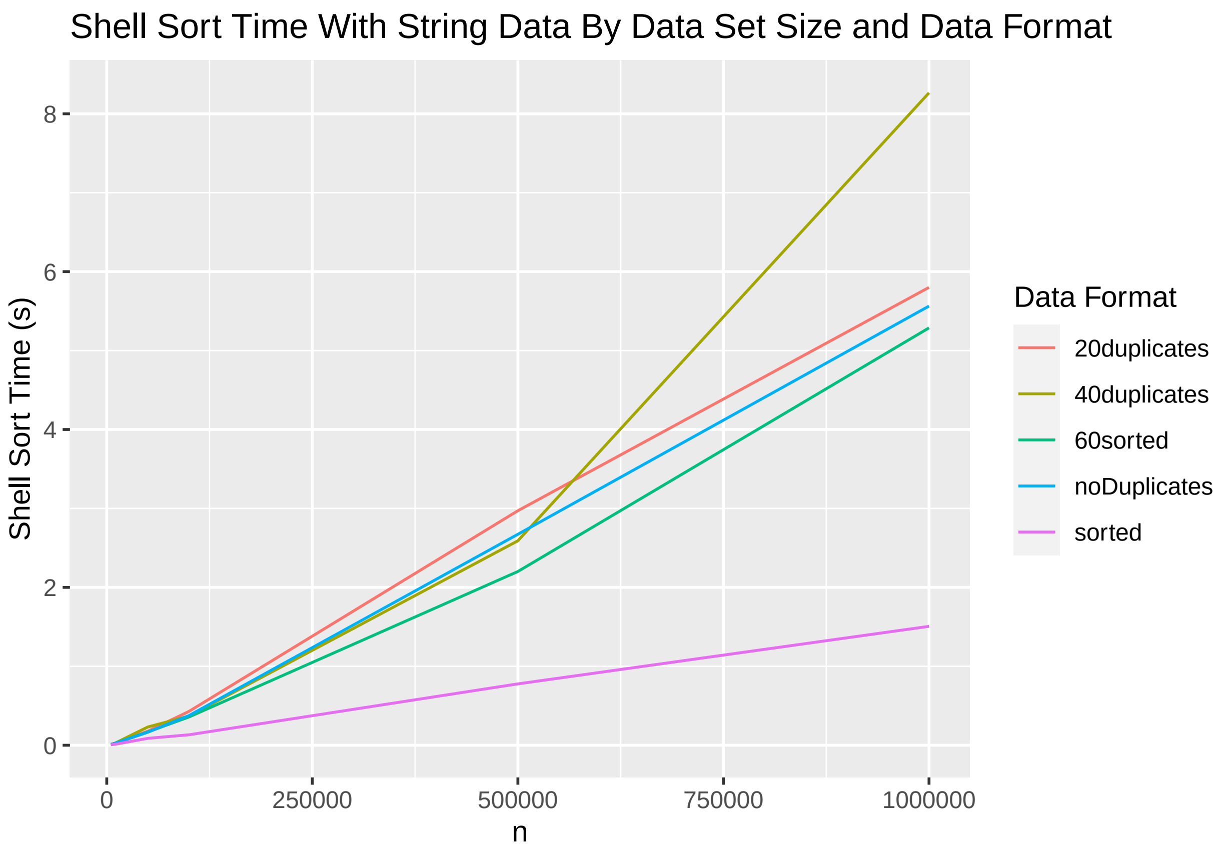
```
shellInts = subset(shellTimes, var_type == "int")
ggplot(shellInts, aes(x = size, y = shell_time, color = format)) +
  geom_line() +
  labs(title = "Shell Sort Time With Integer Data By Data Set Size and Data Format", x = "n", y = "Shel
  guides(color = guide_legend(title = "Data Format"))
```



```r
ggsave("shellInts.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```r
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 Data Format", x = "n", y = "Shell
  guides(color = guide_legend(title = "Data Format"))
```

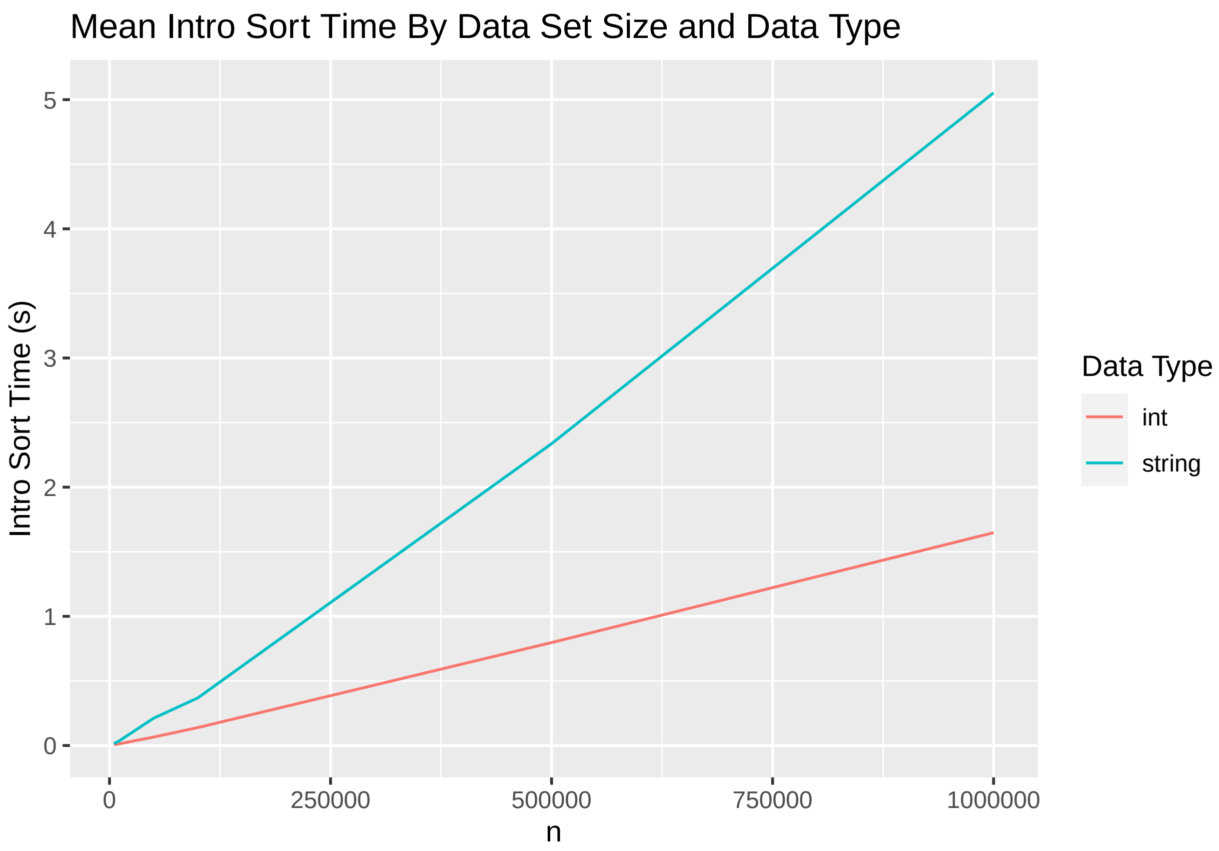


```
ggsave("shellStrings.svg")
```

```
## Saving 6.5 x 4.5 in image
```

### 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 (s)
  guides(color = guide_legend(title = "Data Type"))
```

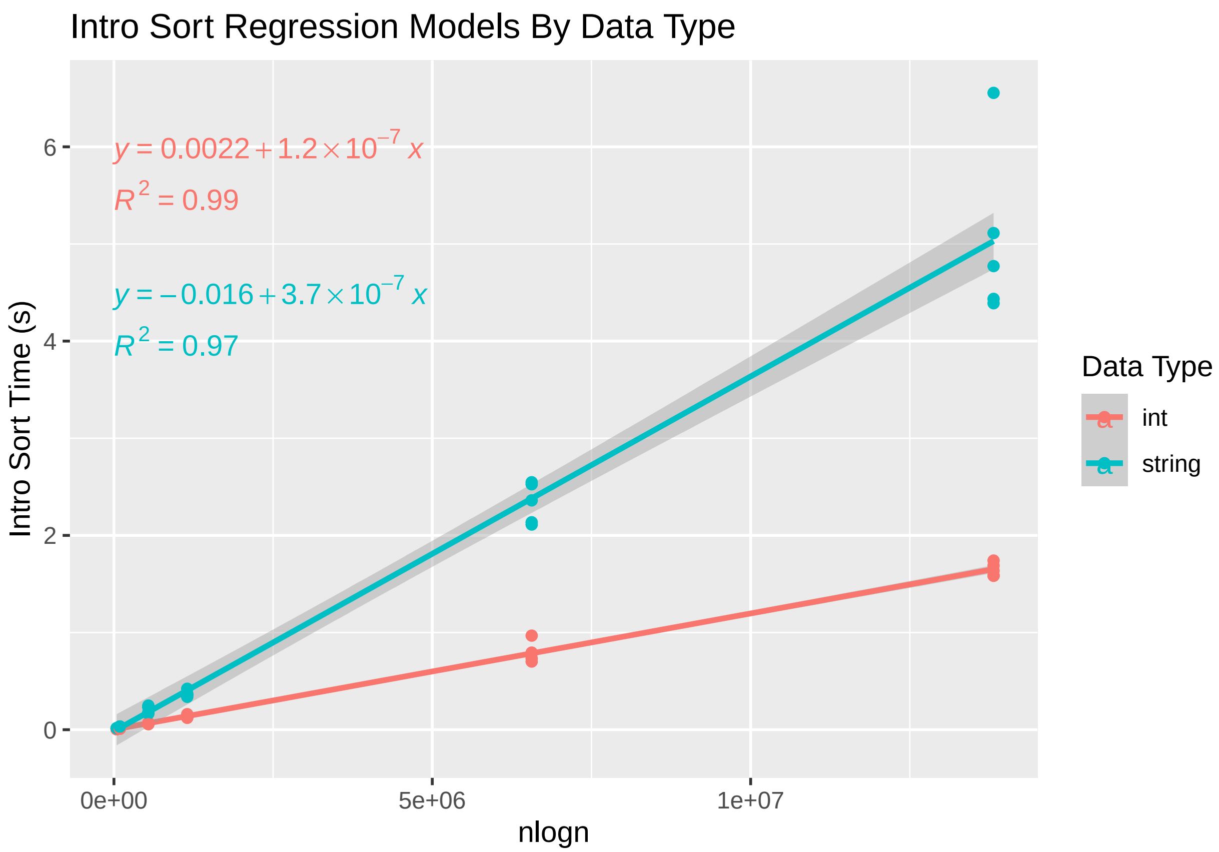


```
ggsave("introMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 (s)") +
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(6, 4.5)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(5.5, 4)) +
  guides(color = guide_legend(title = "Data Type"))
```

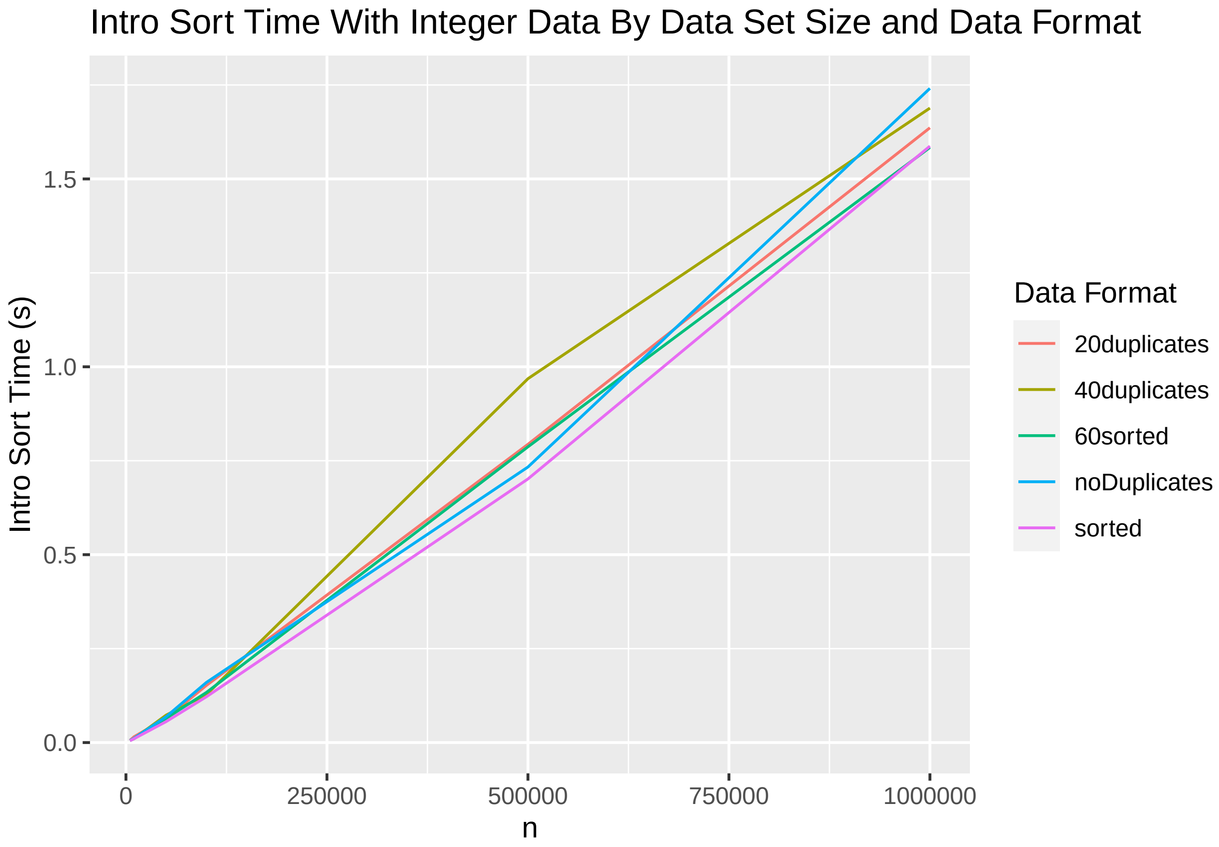
```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



```
ggsave("introRegression.svg")
```

```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using formula 'y ~ x'
```

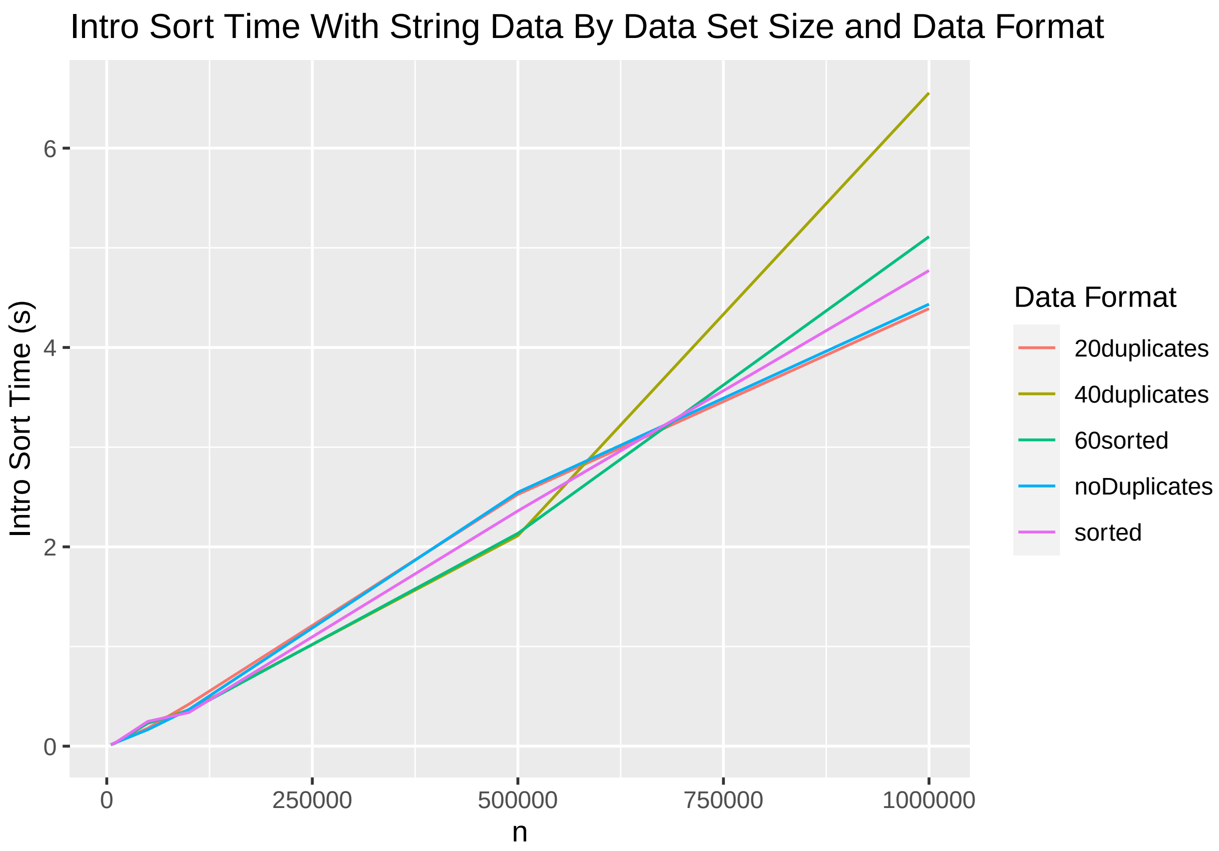
```
introInts = subset(introTimes, var_type == "int")
ggplot(introInts, aes(x = size, y = intro_time, color = format)) +
  geom_line() +
  labs(title = "Intro Sort Time With Integer Data By Data Set Size and Data Format", x = "n", y = "Intro
  guides(color = guide_legend(title = "Data Format"))
```



```
ggsave("introInts.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 Data Format", x = "n", y = "Intro
  guides(color = guide_legend(title = "Data Format"))
```

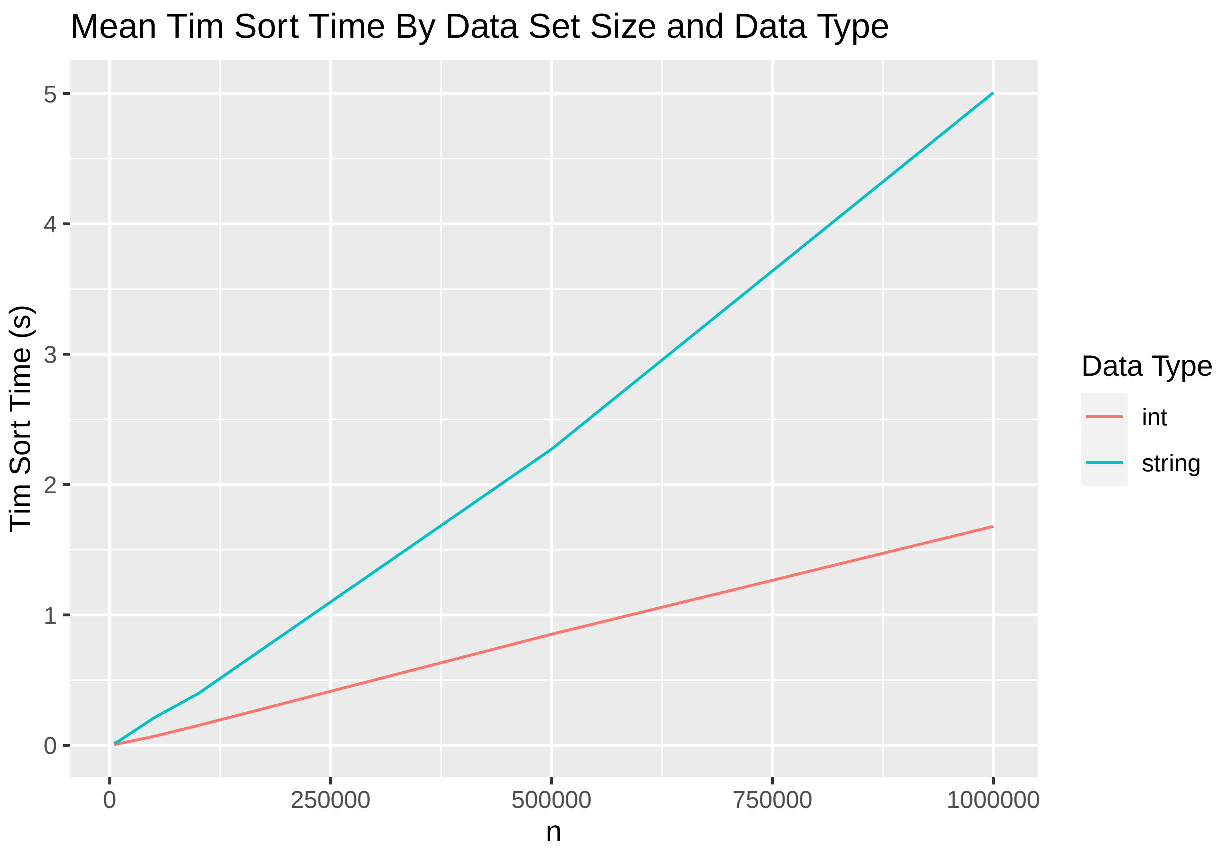


```
ggsave("introStrings.svg")
```

```
## Saving 6.5 x 4.5 in image
```

### 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 (s)") +
  guides(color = guide_legend(title = "Data Type"))
```

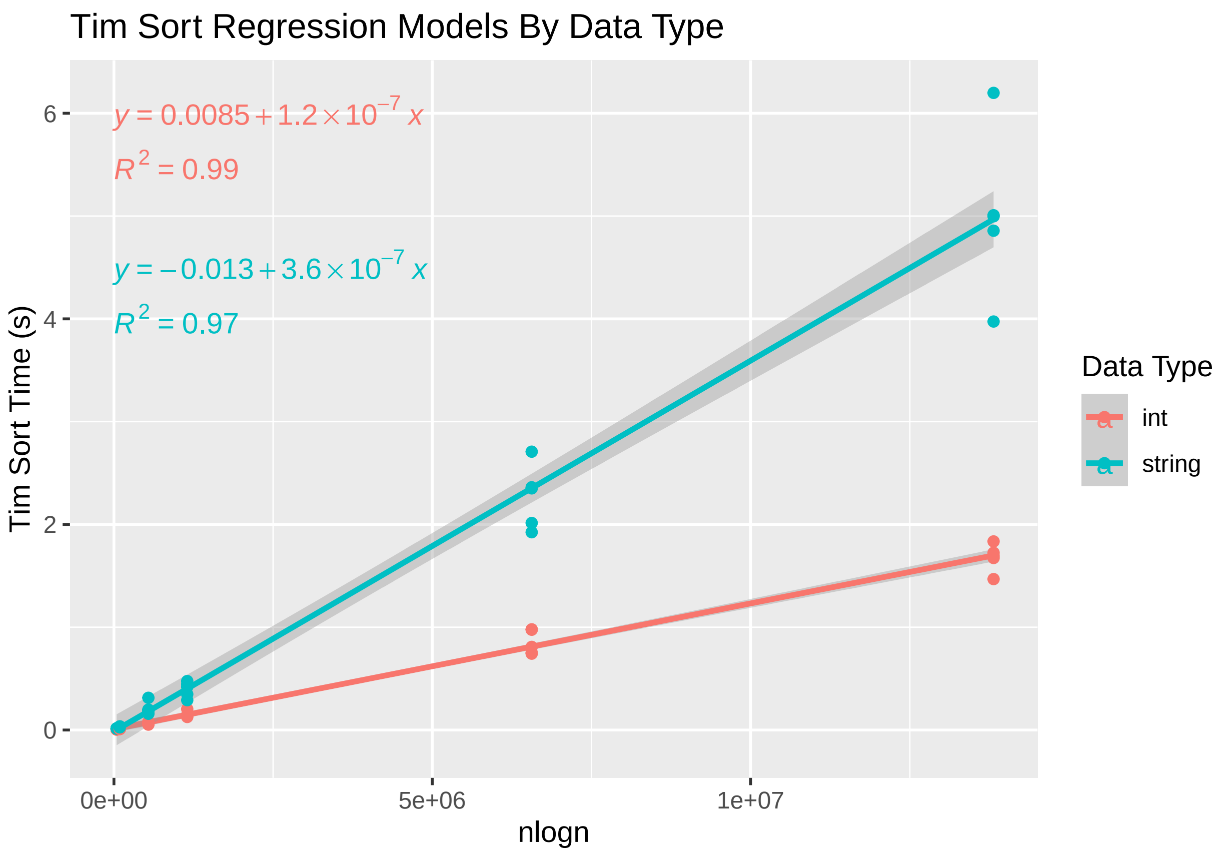


```
ggsave("timMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
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 (s)") +
  geom_smooth(method="lm") +
  geom_point() +
  stat_regline_equation(label.x=0, label.y=c(6, 4.5)) +
  stat_cor(aes(label=..rr.label..), label.x=0, label.y=c(5.5, 4)) +
  guides(color = guide_legend(title = "Data Type"))
```

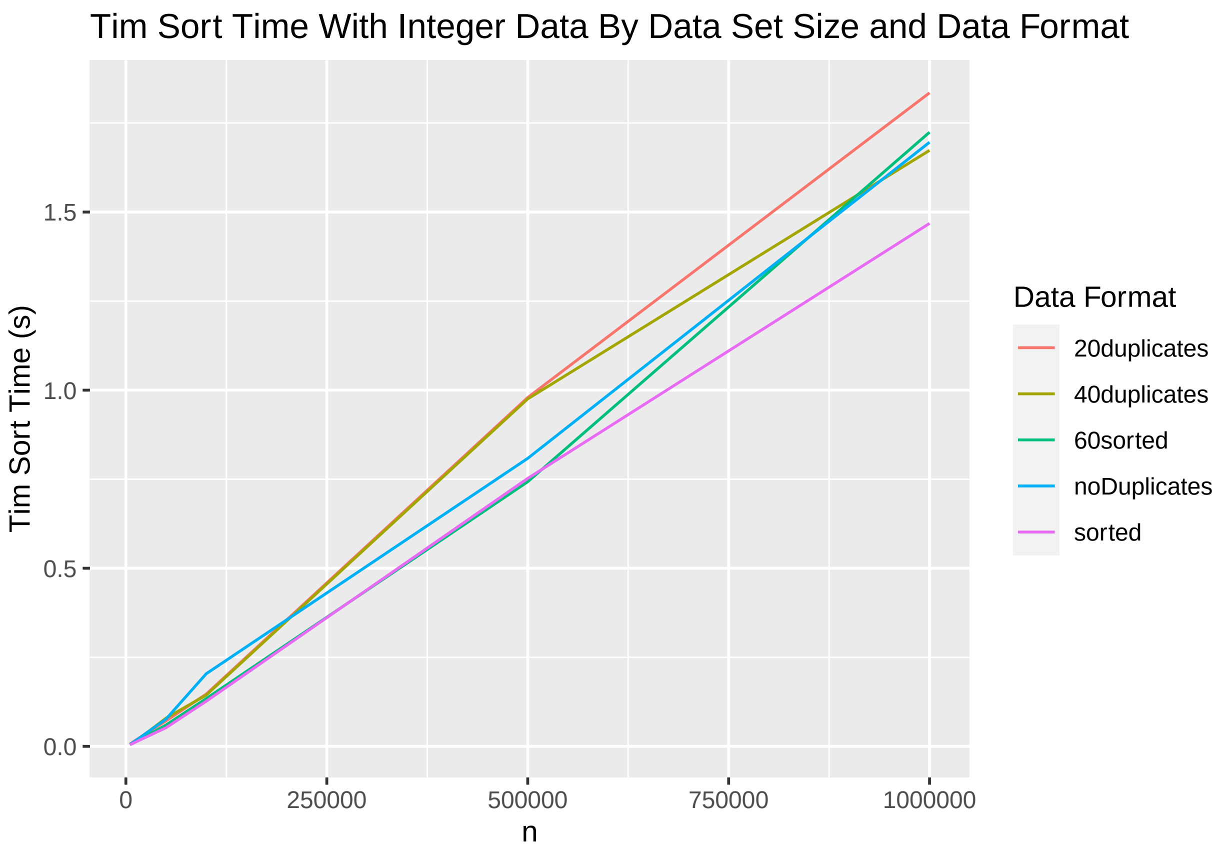
```
## `geom_smooth()` using formula 'y ~ x'
```



```
ggsave("timRegression.svg")
```

```
## Saving 6.5 x 4.5 in image
## ‘geom_smooth()‘ using formula ’y ~ x’
```

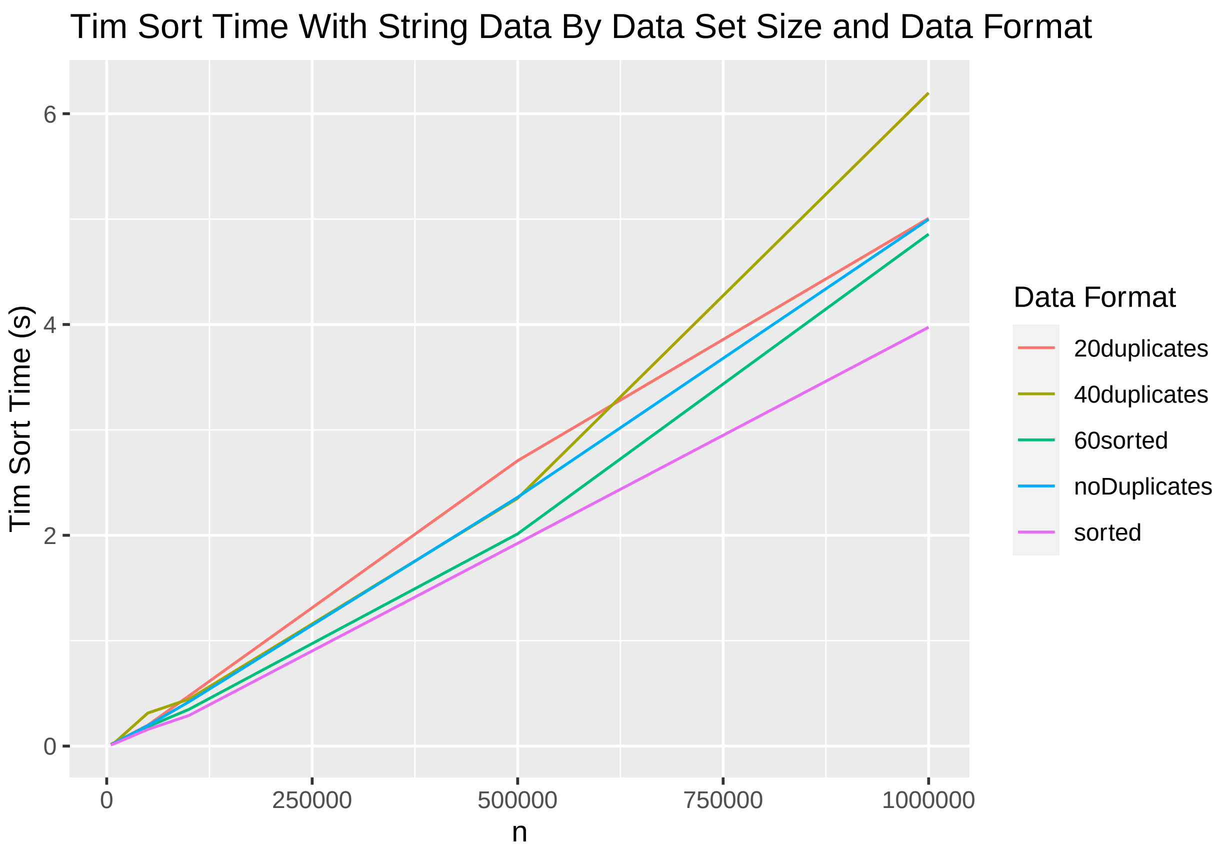
```
timInts = subset(timTimes, var_type == "int")
ggplot(timInts, aes(x = size, y = tim_time, color = format)) +
  geom_line() +
  labs(title = "Tim Sort Time With Integer Data By Data Set Size and Data Format", x = "n", y = "Tim So
  guides(color = guide_legend(title = "Data Format"))
```



```r
ggsave("timInts.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```r
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 Data Format", x = "n", y = "Tim Sor
  guides(color = guide_legend(title = "Data Format"))
```



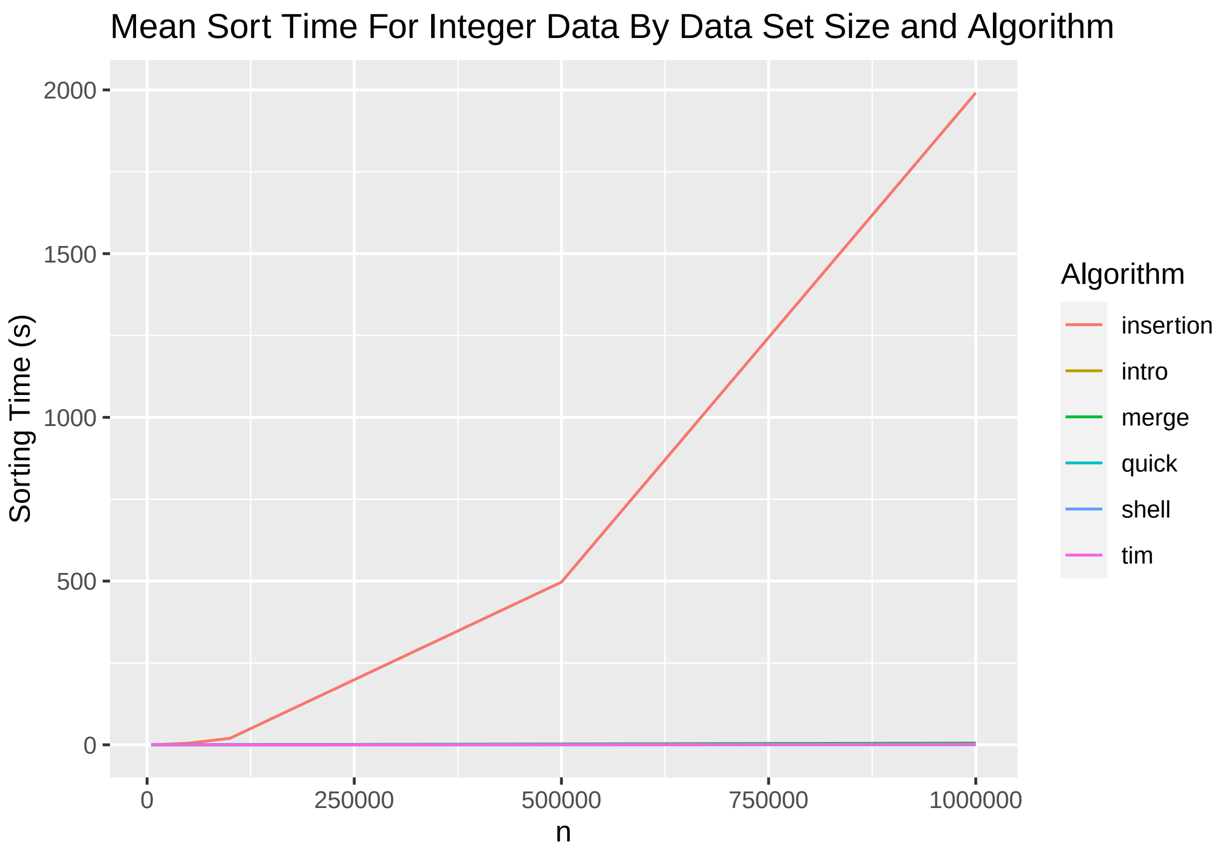
```
ggsave("timStrings.svg")
```

```
## Saving 6.5 x 4.5 in image
```

### Algorithm Comparison

```
# Create new data frame that is restructured to have 1 time column
data2 = matrix(ncol = 5, nrow = 360)
row = 1
# For each algorithm:
for (i in 1:6) {
  # For each input file:
  for (j in 1:60) {
    # Set the first 3 cols to the corresponding cols from old dataset
    data2[row, 1] = data[j, 1]
    data2[row, 2] = data[j, 2]
    data2[row, 3] = data[j, 3]
    # Set the time from the correct algorithm
    data2[row, 4] = data[j, 3 + i]
    # Set which algorithm this row is from
    if (i == 1) {
      data2[row, 5] = "insertion"
    } else if (i == 2) {
      data2[row, 5] = "quick"
    } else if (i == 3) {
      data2[row, 5] = "merge"
    } else if (i == 4) {
      data2[row, 5] = "shell"
    } else if (i == 5) {
      data2[row, 5] = "intro"
    } else {
      data2[row, 5] = "tim"
    }
    row = row + 1
  }
}
data2 = data.frame(data2)
# Set the column names
colnames(data2) = c("var_type", "size", "format", "time", "algorithm")
# Convert numeric columns back into numeric data
data2 = transform(data2, time = as.numeric(time))
data2 = transform(data2, size = as.numeric(size))
```

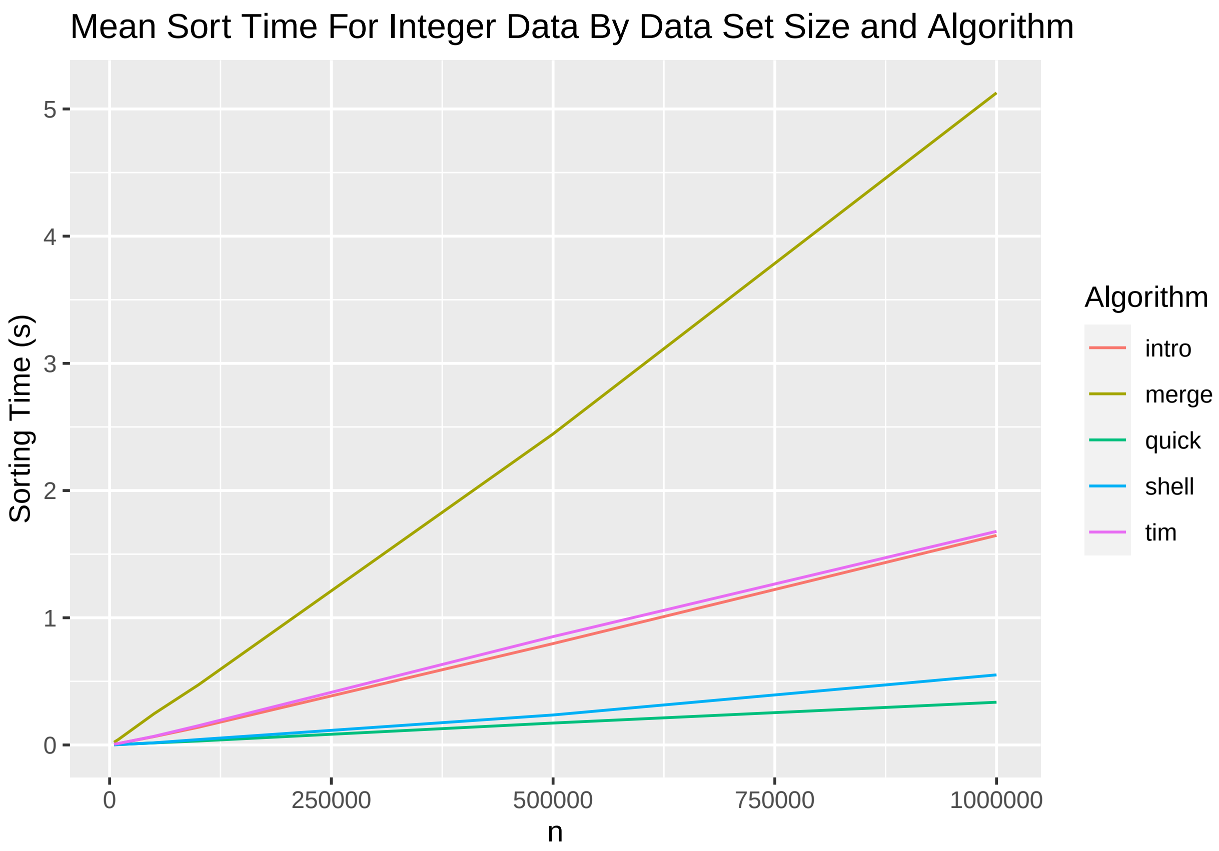
```
integerData = subset(data2, var_type == "int")
integerTimes = aggregate(time ~ algorithm + size, data = integerData, FUN = mean)
ggplot(integerTimes, aes(x = size, y = time, color = algorithm)) +
  geom_line() +
  labs(title = "Mean Sort Time For Integer Data By Data Set Size and Algorithm", x = "n", y = "Sorting T
  guides(color = guide_legend(title = "Algorithm"))
```



```
ggsave("intMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

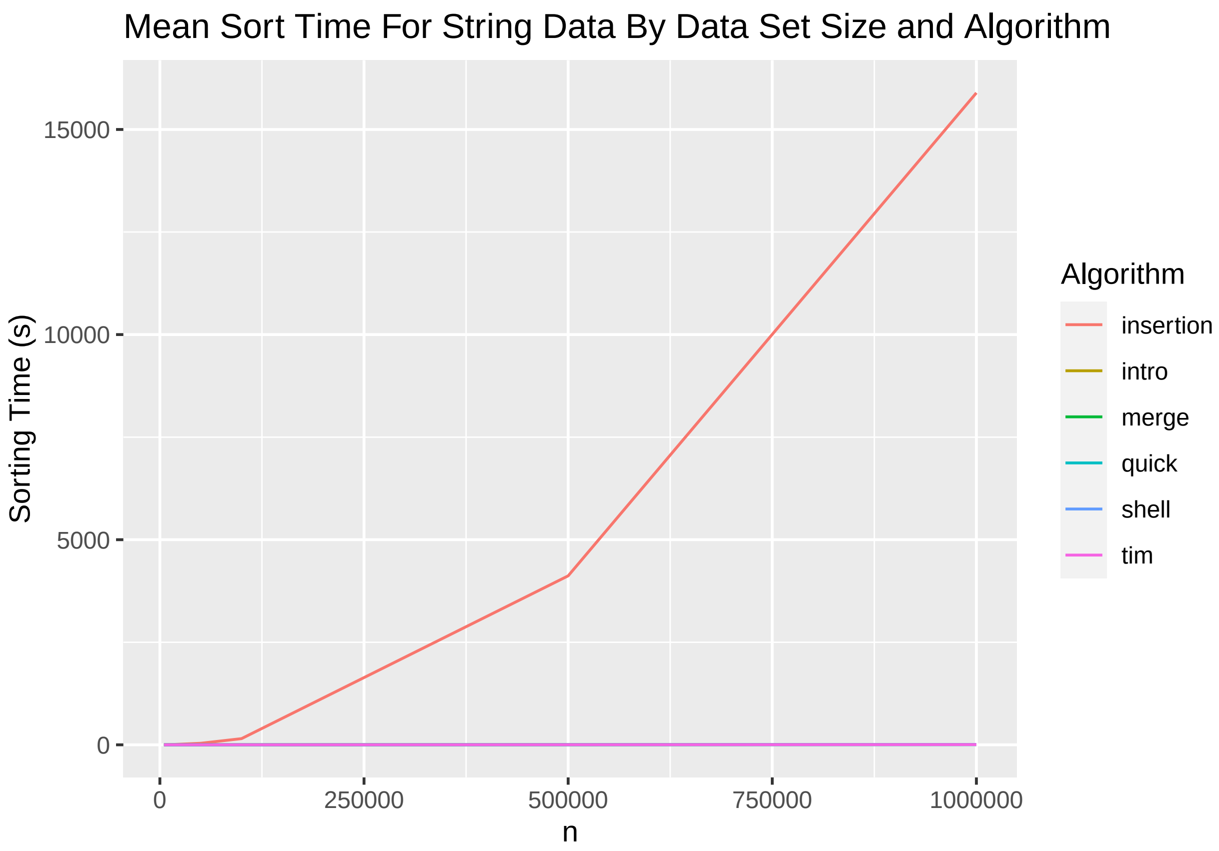
```
integerTimes2 = subset(integerTimes, algorithm != "insertion")
ggplot(integerTimes2, aes(x = size, y = time, color = algorithm)) +
  geom_line() +
  labs(title = "Mean Sort Time For Integer Data By Data Set Size and Algorithm", x = "n", y = "Sorting
  guides(color = guide_legend(title = "Algorithm"))
```



```
ggsave("intMean2.svg")
```

```
## Saving 6.5 x 4.5 in image
```

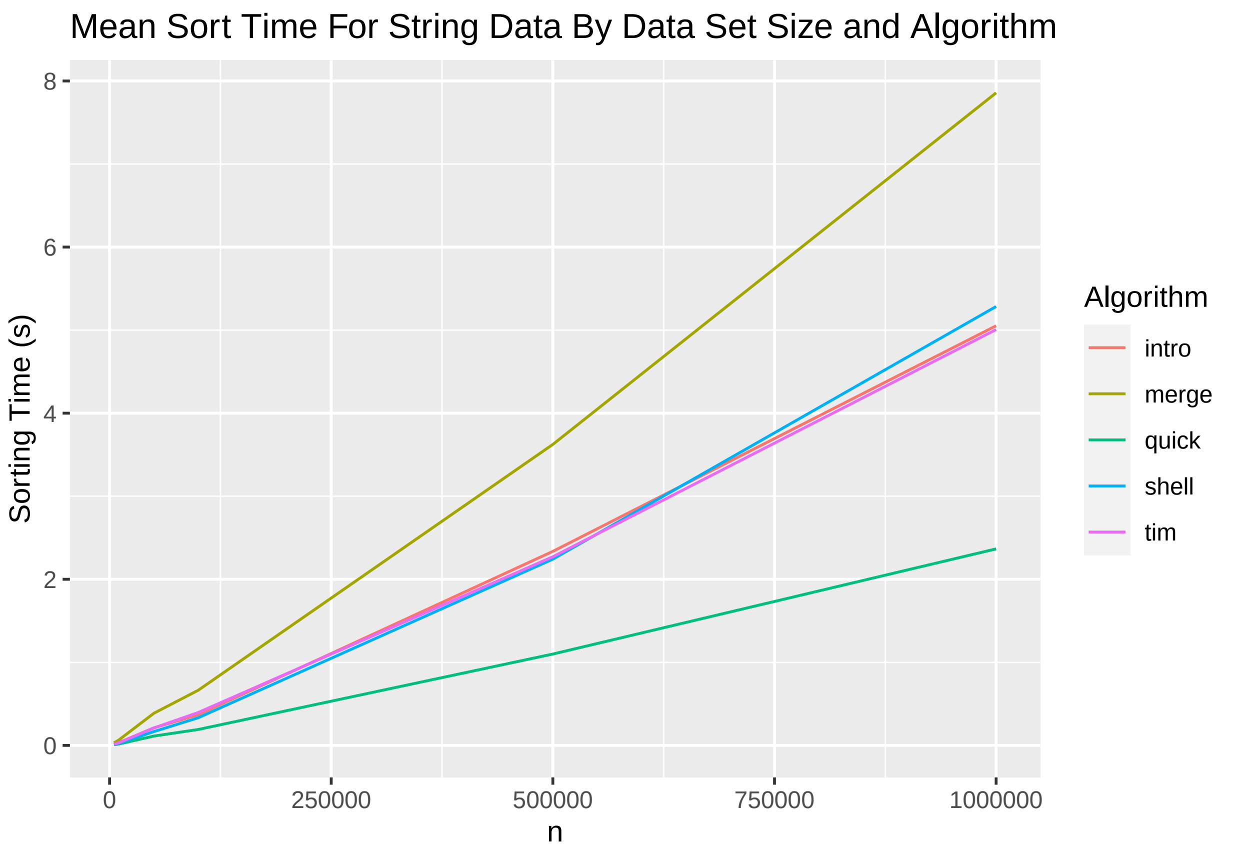
```
stringData = subset(data2, var_type == "string")
stringTimes = aggregate(time ~ algorithm + size, data = stringData, FUN = mean)
ggplot(stringTimes, aes(x = size, y = time, color = algorithm)) +
  geom_line() +
  labs(title = "Mean Sort Time For String Data By Data Set Size and Algorithm", x = "n", y = "Sorting Ti
  guides(color = guide_legend(title = "Algorithm"))
```



```r
ggsave("stringMean.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```r
stringTimes2 = subset(stringTimes, algorithm != "insertion")
ggplot(stringTimes2, aes(x = size, y = time, color = algorithm)) +
  geom_line() +
  labs(title = "Mean Sort Time For String Data By Data Set Size and Algorithm", x = "n", y = "Sorting Ti
  guides(color = guide_legend(title = "Algorithm"))
```



```r
ggsave("stringMean2.svg")
```

```
## Saving 6.5 x 4.5 in image
```