Grade Calculator

Byteflow Dynamics

10/1/2017

Goal

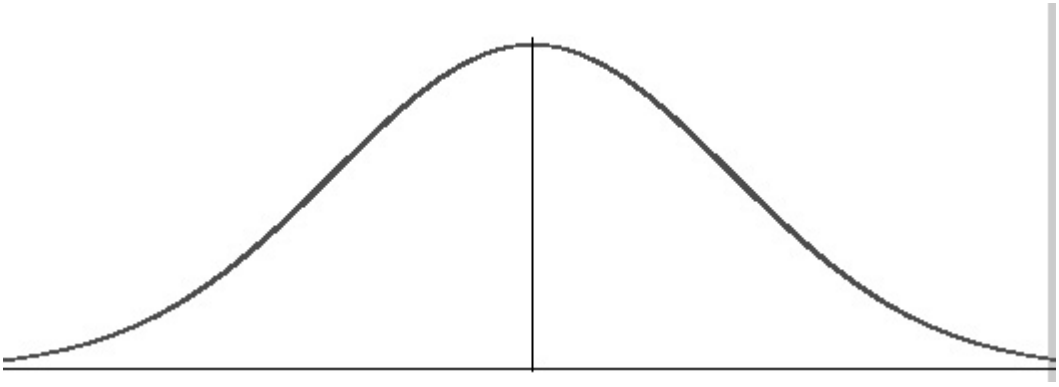
The purpose of the grade calculator is to assign each student a letter grade based on their percent grade. The percent grade is calculated by applying various operations to their raw scores based on the grading policy of the instructor.

For this course, our grading policy is as follows:

- Midterm, Final, Extra credit scores are on a 100 scale, all other scores are on a 10 scale (Extra credit used to supplement exam scores)
- Final grade is based 50% on lecture (exams, quizzes, homework) and 50% on lab
- 2 lowest lab scores are dropped, other labs all weight equally
- Lecture grade is based 30% each on Midterm, Final exam, and Homework, 10% on Quizzes

- Extra credit for lecture is added to the Final exam score on a 100 scale
- Letter grades (A, B, C, D, F) are determined by the norml distribution scheme:
  - A: top 2%
  - B: next 14%
  - C: middle 68%
  - D: next 14%
  - F: bottom 2%

Normal (Gaussian) Distribution

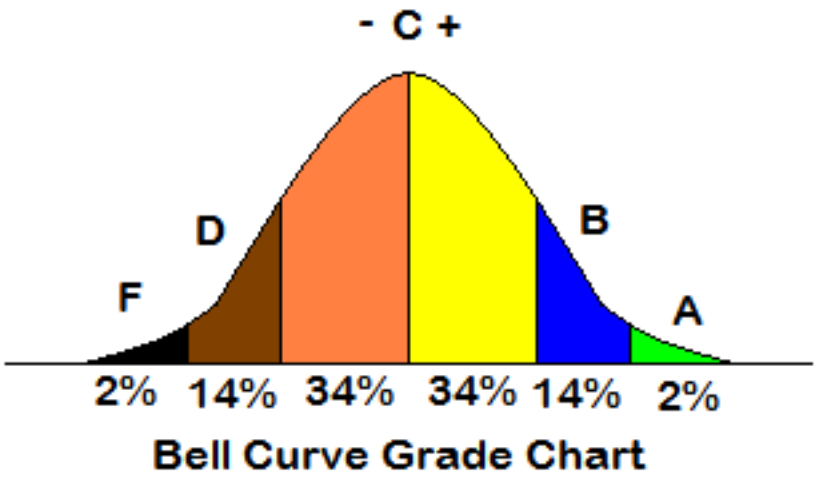


- Shaped like a bell and it's symmetric
- A normal distribution occurs naturally when the sample size is sufficiently large
- In theory, student grades follow a normal distribution

Normal distribution variables

- Mean μ (average)

- Standard deviation σ
  - Measure of how spread out the data values are
  - 68% of all data falls within 1σ of the mean (+/-34% from the mean)
  - 95% of the data falls within 2σ of the mean



$$f(x) = \frac{e^{-(x-\mu)^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}}$$

Question: If the mean test score is 65 and the standard deviation is 10, what letter grade would a score 80 get? (2 min)

Answer:

$\mu = 65$, $\sigma = 10$

$\mu + 1\sigma = 75$: C upper bound

$\mu + 2\sigma = 85$: B upper bound

–> B

How to tackle a grade calculator

There are multiple operations to be executed in order to calculate final grades. Let's break down the dataset into pieces and apply appropriate functions to each piece.

Things to think about

- What operations are needed for which part of data
- In what order you should proceed
- Which scale to adjust to (normally to a 100 scale)

Add a new variable "grade_type"

grade_type variable shows if it's a lab, quiz, homework, etc. This makes it easier to group only lab grades, homework grades, etc.

```
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score
## 1   1 1         A          T Lab.1     0
## 2   2 2         B          M Lab.1     0
## 3   3 3         B          J Lab.1    10
## 4   4 4         B          B Lab.1    10
## 5   5 5         C          M Lab.1    10
## 6   6 6         C          S Lab.1    10
```

```
scores <- scores %>%
  mutate(grade_type = sub("\\d+","", type)) %>% # this line alone is sufficient to remove digits
  mutate(grade_type = sub(".Ch.","", grade_type)) %>% # just to make it prettier
  mutate(grade_type = sub(".ch.","", grade_type)) %>%
  mutate(grade_type = sub("Lab.", "Lab", grade_type))
head(scores)
```

```
##   X.1 X Last.Name First.Name  type score grade_type
## 1   1 1         A          T Lab.1     0        Lab
## 2   2 2         B          M Lab.1     0        Lab
## 3   3 3         B          J Lab.1    10        Lab
## 4   4 4         B          B Lab.1    10        Lab
## 5   5 5         C          M Lab.1    10        Lab
## 6   6 6         C          S Lab.1    10        Lab
```

Calculate lecture grade

Lecture grade rules

- Lecture grade is based 30% each on Midterm, Final exam, and Homework, 10% on Quizzes
- Extra credit for lecture is added to the Final exam score on a 100 scale

Steps

1. Calculate weighted exam grade including Extra credit
2. Calculate weighted homework and quiz grades
3. Combine

Calculate weighted exam grade

- Exam grade (Midterm and Final exam) is worth 60% of lecture grade
- Since Midterm and Final exam weight equally, we can add these scores and Extra credit together and scale
- Exams are out of 100

Steps to calculate exam grade

1. Group by student names
2. Filter midterm, final, and extra credit scores
3. Summarise to add 3 scores and weight

```
scores_exam <- scores %>%
  group_by(Last.Name, First.Name) %>%
  filter(type == "Midterm" | type == "Final.exam" | type == "Extra.Credit") %>%
  summarise(exam_weighted = sum(score) * 0.3) # 30% weight on midterm and final
head(scores_exam)
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name exam_weighted
##       <chr>      <chr>         <dbl>
## 1         A          T          30.3
## 2         B          B          50.1
## 3         B          J          59.7
## 4         B          M           0.0
## 5         C          L          42.3
## 6         C          M          44.4
```

Exercise: Calculate weighted exam grades if final grades are based 20% on Midterm and 30% on Final exam. (5 min)

Hint: Make 2 intermediate data frames (for midterm and final exam) and combine them

---

1. Group by names
2. Filter Midterm/Final separately
3. Summarise to weight
4. Combine two data frames (see cheat sheet)

```
mt <- scores %>%
  group_by(Last.Name, First.Name) %>%
  filter(type == "Midterm") %>%
  summarise(weighted_midterm = score * 0.2)

fe <- scores %>%
  group_by(Last.Name, First.Name) %>%
  filter(type == "Final.exam") %>%
  summarise(weighted_final = score * 0.3)

exam <- inner_join(mt, fe) %>%
  mutate(weighted_exam = weighted_midterm + weighted_final)
```

```
## Joining, by = c("Last.Name", "First.Name")
```

```
head(exam)
```

```
## # A tibble: 6 x 5
## # Groups:   Last.Name [3]
##   Last.Name First.Name weighted_midterm weighted_final weighted_exam
```

```
##       <chr>      <chr>            <dbl>          <dbl>         <dbl>
## 1         A          T              8.0           11.7          19.7
## 2         B          B             13.6           20.7          34.3
## 3         B          J             14.4           26.1          40.5
## 4         B          M              0.0            0.0           0.0
## 5         C          L             13.8           21.6          35.4
## 6         C          M             16.2           20.1          36.3
```

Calculate weighted Homework and Quiz grade

- Lecture grade is based 30% each on Midterm, Final exam, and Homework, 10% on Quizzes
- Homework and Quizzes are out of 10

Steps to calculate weighed homework grade

1. Group by names
2. Filter HW
3. Summarise to average scores and weight (Homework worth 30%)

```
scores_hw <- scores %>%
  group_by(Last.Name, First.Name) %>%
  filter(grade_type == "HW") %>%
  summarise(hw_weighted = mean(score) * 3) #multiply by 3 to make it out of 30
head(scores_hw)
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name hw_weighted
##       <chr>      <chr>       <dbl>
## 1         A          T        13.8
## 2         B          B        18.9
## 3         B          J        25.2
## 4         B          M         0.0
## 5         C          L        20.1
## 6         C          M        21.6
```

Exercise: Calculate weighted quiz grade (3 min)

---

1. Group by names
2. Filter Quiz
3. Summarise to average scores and weight (Quiz worth 10%)

```
scores_quiz <- scores %>%
  group_by(Last.Name, First.Name) %>%
  filter(grade_type == "Quiz") %>%
  summarise(quiz_weighted = mean(score)) # quiz scores are already out of 10 so no need to multiply wit
head(scores_quiz)
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name quiz_weighted
```

```
##       <chr>      <chr>         <dbl>
## 1         A          T          6.75
## 2         B          B          7.40
## 3         B          J          8.90
## 4         B          M          0.00
## 5         C          L          5.75
## 6         C          M          8.50
```

Combine Exam, Homework, and Quiz grades

1. Use inner_join to combine 3 data frames
2. Use mutate to add a column containing sum of 3 grades

```
scores_lec <- inner_join(scores_exam, scores_hw) %>%
  inner_join(scores_quiz) %>%
  mutate(lec_grade = exam_weighted + hw_weighted + quiz_weighted)
```

```
## Joining, by = c("Last.Name", "First.Name")
## Joining, by = c("Last.Name", "First.Name")
```

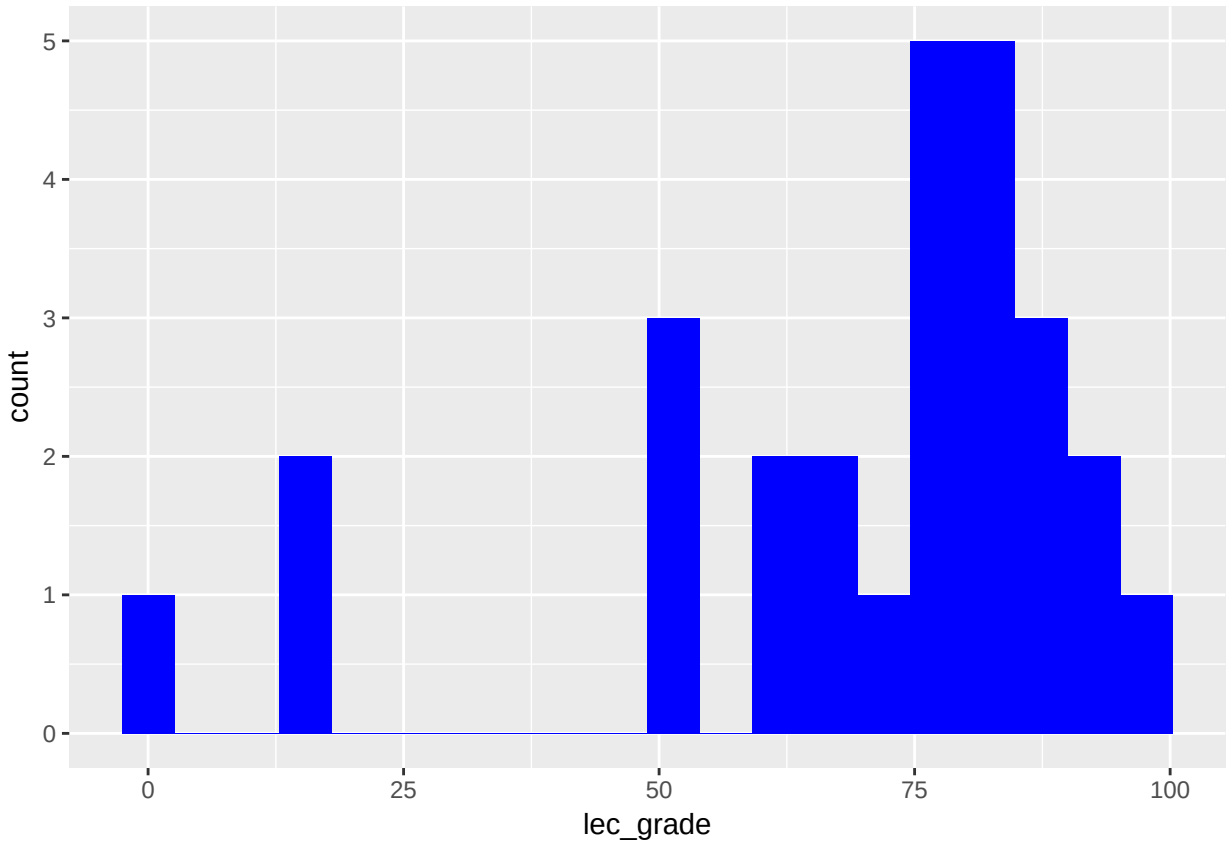
```
head(scores_lec)
```

```
## # A tibble: 6 x 6
## # Groups:   Last.Name [3]
##   Last.Name First.Name exam_weighted hw_weighted quiz_weighted lec_grade
##       <chr>      <chr>         <dbl>       <dbl>         <dbl>     <dbl>
## 1         A          T          30.3        13.8          6.75     50.85
## 2         B          B          50.1        18.9          7.40     76.40
## 3         B          J          59.7        25.2          8.90     93.80
## 4         B          M           0.0         0.0          0.00      0.00
## 5         C          L          42.3        20.1          5.75     68.15
## 6         C          M          44.4        21.6          8.50     74.50
```

Do some statistics and visualization

Plot a histogram

```
ggplot(data = scores_lec) +
  geom_histogram(mapping = aes(x = lec_grade), bins =20, fill = "blue")
```



Calculate quantiles

Quantiles are used to divide a distribution into sub-groups.

Remember a normal distribution,

We need to find [2%, 16%, 50%, 84%, 98%] quantiles to see what grade corresponds to each letter grade.

The **summary** function gives you some statistics including mean, min & max, and quartiles

```
summary(scores_lec$lec_grade)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   62.17   78.55   69.04   83.38   97.70
```

To find specific quantiles, use **quantile** function

```
quantile(scores_lec$lec_grade, c(0.02,0.16,0.5,0.84,0.98))
```

```
##     2%    16%    50%    84%    98%
##  6.734 51.322 78.550 86.642 95.932
```

We can use these to assign letter grades

Exercise: Divide the same lecture grade data into 5 equal-sized groups using quantile (2 min)

```
quantile(scores_lec$lec_grade, c(0.2,0.4,0.6,0.8))
```

```
##   20%   40%   60%   80%
## 55.08 74.84 80.38 85.98
```

Calculate lab grade

- 2 lowest lab scores are dropped, other labs all weight equally
- To drop lowest labs, we need to rank lab scores

Steps

1. Group by student names
2. Rank lab scores (Use a mutate function row_number)
3. Drop 2 lowest scores
4. Summarise remaining scores and adjust to a 100 scale

Rank lab scores

```
lab_ranked <- scores %>%
  group_by(Last.Name, First.Name) %>%
  filter(grade_type == "Lab") %>%
  mutate(rank = row_number(score)) %>%
  arrange(Last.Name, First.Name, rank) # to check the ranking function is working

head(lab_ranked)
```

```
## # A tibble: 6 x 8
## # Groups:   Last.Name, First.Name [1]
##     X.1     X Last.Name First.Name  type score grade_type  rank
##   <int> <int>     <chr>      <chr> <chr> <dbl>      <chr> <int>
## 1     1     1         A          T Lab.1     0        Lab     1
## 2    82     1         A          T Lab.4     0        Lab     2
## 3   109     1         A          T Lab.5     0        Lab     3
## 4   163     1         A          T Lab.7     0        Lab     4
## 5   190     1         A          T Lab.8     0        Lab     5
## 6   217     1         A          T Lab.9     0        Lab     6
```

Drop 2 lowest labs

```
lab_dropped <- lab_ranked %>%
  filter(rank > 2)
head(lab_dropped)
```

```
## # A tibble: 6 x 8
## # Groups:   Last.Name, First.Name [1]
##     X.1     X Last.Name First.Name   type score grade_type  rank
##   <int> <int>     <chr>      <chr>  <chr> <dbl>      <chr> <int>
## 1   109     1         A          T  Lab.5     0        Lab     3
## 2   163     1         A          T  Lab.7     0        Lab     4
```

```
## 3   190     1         A          T  Lab.8     0        Lab     5
## 4   217     1         A          T  Lab.9     0        Lab     6
## 5   271     1         A          T Lab.11     0        Lab     7
## 6   298     1         A          T Lab.12     0        Lab     8
```

Summarise remaining lab scores

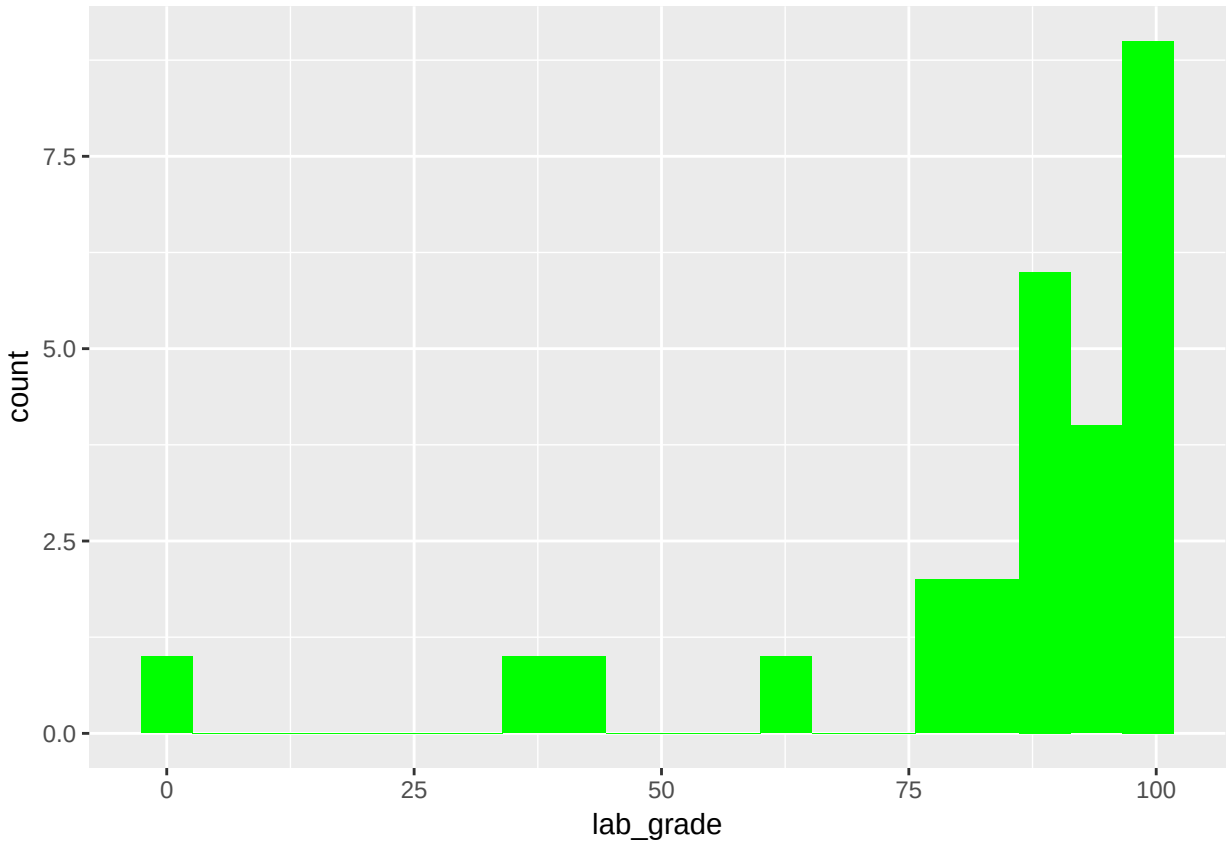
```
scores_lab <- lab_dropped %>%
  summarise(lab_grade = mean(score) * 10) # adjust to a 100 scale

head(scores_lab)
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name lab_grade
##       <chr>      <chr>     <dbl>
## 1         A          T  35.83333
## 2         B          B  97.91667
## 3         B          J  99.16667
## 4         B          M   0.00000
## 5         C          L  97.50000
## 6         C          M  80.00000
```

Statistics and visualization

Plot a histogram

```
ggplot(data = scores_lab) +
  geom_histogram(mapping = aes(x = lab_grade), bins =20, fill = "green")
```



Calculate quantiles

```
quantile(scores_lab$lab_grade, c(0.02,0.16,0.5,0.84,0.98))
```

```
##       2%      16%      50%      84%      98%
## 18.63333 79.30000 90.83333 97.85000 98.73333
```

Combine lab and lecture grades

- Final grade is based 50% on lecture and 50% on lab

Steps

1. Use inner_joing to combine lecture and lab data frames
2. Use mutate to add a column containing final grade
3. Drop unnecessary columns

```
final_grade <- inner_join(scores_lab, scores_lec) %>%
  mutate(grade_all = (lec_grade + lab_grade)/2) %>%
  select(Last.Name, First.Name, grade_all)
```

```
## Joining, by = c("Last.Name", "First.Name")
```

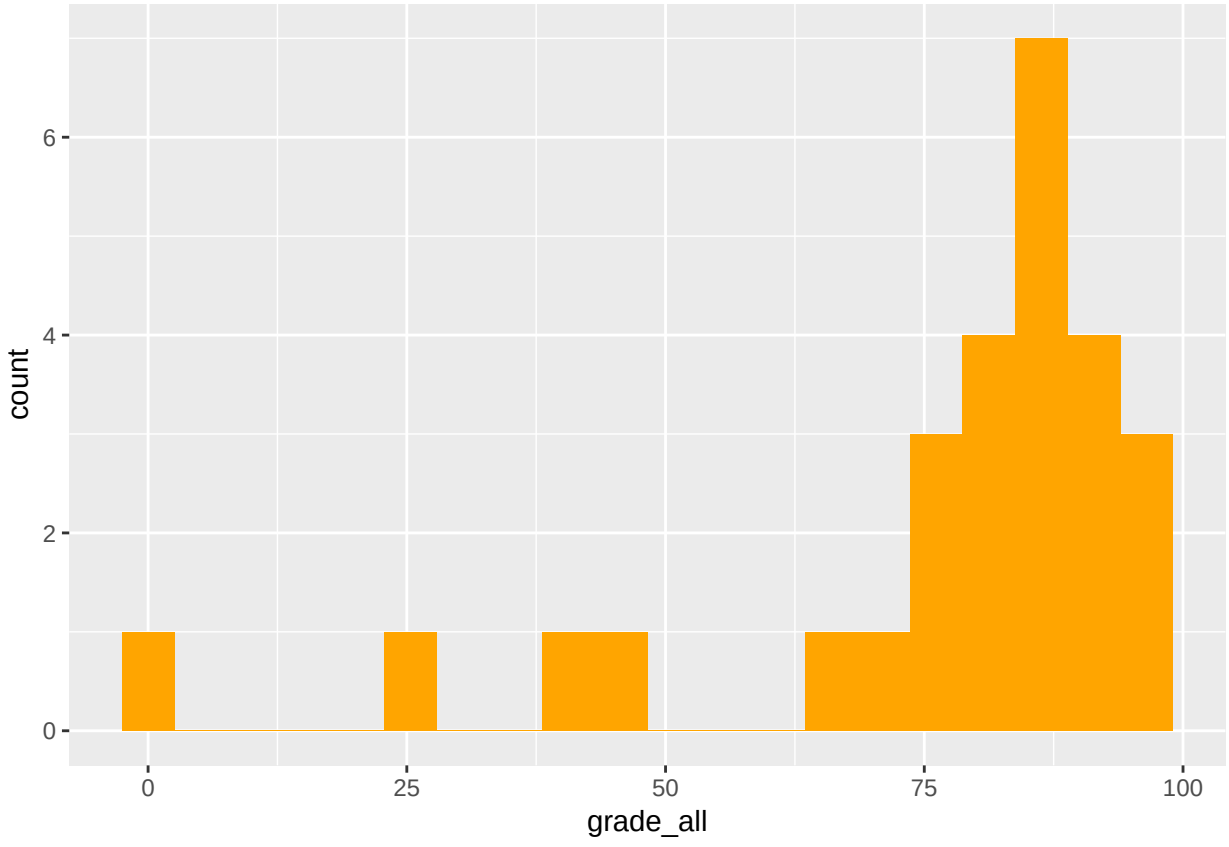
```
head(final_grade)
```

```
## # A tibble: 6 x 3
## # Groups:   Last.Name [3]
##   Last.Name First.Name grade_all
##       <chr>      <chr>     <dbl>
## 1         A          T  43.34167
## 2         B          B  87.15833
## 3         B          J  96.48333
## 4         B          M   0.00000
## 5         C          L  82.82500
## 6         C          M  77.25000
```

Statistics and visualization

Plot a histogram

```
ggplot(data = final_grade) +
  geom_histogram(mapping = aes(x = grade_all), bins =20, fill = "orange")
```



Calculate quantiles

```
quantile(final_grade$grade_all, c(0.02,0.16,0.5,0.84,0.98))
```

```
##       2%      16%      50%      84%      98%
## 13.55033 68.01867 86.24167 91.76500 96.41400
```

Now assign letter grades

From the quantiles, we have

A: above 96.4

B: between 91.8 & 96.4

C: between 68.0 & 91.7

D: between 13.6 & 68.0

F: below 13.6

Steps to assign letter grades

1. Create an intermediate data frame for each letter grade by copying final_grade data frame
2. Filter rows with percent grades corresponding to each letter grade
3. Combine 5 data frames by binding them by rows (Use bind_rows)
4. Arrange to reorder by names

```
grade_A <- final_grade %>%
  filter(grade_all >= 96.4) %>%
  mutate(grade = "A")

grade_B <- final_grade %>%
  filter(grade_all >= 91.8 & grade_all < 96.4) %>%
  mutate(grade = "B")

grade_C <- final_grade %>%
  filter(grade_all >= 68 & grade_all < 91.8) %>%
  mutate(grade = "C")

grade_D <- final_grade %>%
  filter(grade_all >= 13.6 & grade_all < 68) %>%
  mutate(grade = "D")

grade_F <- final_grade %>%
  filter(grade_all < 13.6) %>%
  mutate(grade = "F")

letter_grade <- bind_rows(grade_A, grade_B) %>%
  bind_rows(grade_C) %>%
  bind_rows(grade_D) %>%
  bind_rows(grade_F) %>%
  arrange(Last.Name, First.Name)

letter_grade
```

```
## # A tibble: 27 x 4
## # Groups:   Last.Name [14]
##    Last.Name First.Name grade_all grade
```

```
##        <chr>      <chr>     <dbl> <chr>
##  1         A          T  43.34167     D
##  2         B          B  87.15833     C
##  3         B          J  96.48333     A
##  4         B          M   0.00000     F
##  5         C          L  82.82500     C
##  6         C          M  77.25000     C
##  7         C          S  87.85833     C
##  8         C          T  86.24167     C
##  9         D          C  88.32500     C
## 10         D          R  81.24167     C
## # ... with 17 more rows
```