RMHI/ARMP Problem Set 1

Koquiun Li Lin 1319881 [Word Count: 1142]

Please put your answers here, following the instructions in the assignment description. Put your answers and word count tallies in the locations indicated; if none is indicated that means there is no word count for that question. Remember to knit as you go, and submit the knitted version of this on Canvas.

Q1

Q1a

```
# Put your code here
table(d$level, d$talent)
```

```
##
##                comedy dancing instrument magic other singing
##   fun               3       4          3     2     2       8
##   competitive       5       3          3     4     0       5
```

Q1b

```
# Put your code here
d$talent <- factor(d$talent, levels=c("singing","dancing","instrument",
"comedy","magic","other"))
table(d$talent)
```

```
##
##    singing    dancing instrument     comedy      magic      other
##         13          7          6          8          6          2
```

ANSWER: The most common talent was singing, with 13 performances.

Q1c

```
# Put your code here
colnames(d)[2] <- "species"
head(d)
```

```
## # A tibble: 6 × 6
##   name   species level       talent  audience judge
##   <fct>  <fct>   <fct>       <fct>      <dbl> <dbl>
## 1 gladly bear    competitive singing        8     2
## 2 gladly bear    fun         dancing        9    NA
## 3 panda  bear    fun         dancing        8    NA
## 4 snowy  bear    fun         singing        7    NA
## 5 bunny  bunny   competitive magic           8     2
## 6 bunny  bunny   fun         comedy        10    NA
```

Q2

Q2a

```
# Put your code here
d[(d$judge==1 | d$judge==2) & d$audience>=8,]
```

```
## # A tibble: 26 × 6
##    name        species level       talent  audience judge
##    <fct>       <fct>   <fct>       <fct>      <dbl> <dbl>
##  1 gladly      bear    competitive singing        8     2
##  2 <NA>        <NA>    <NA>        <NA>          NA    NA
##  3 <NA>        <NA>    <NA>        <NA>          NA    NA
##  4 bunny       bunny   competitive magic          8     2
##  5 <NA>        <NA>    <NA>        <NA>          NA    NA
##  6 cuddly paws bunny   competitive dancing        8     1
##  7 <NA>        <NA>    <NA>        <NA>          NA    NA
##  8 flopsy      bunny   competitive singing       10     1
##  9 <NA>        <NA>    <NA>        <NA>          NA    NA
## 10 <NA>        <NA>    <NA>        <NA>          NA    NA
## # ℹ 16 more rows
```

Q2b

```
# Put your code here
d %>%
  filter((judge==1 | judge==2),
         audience>=8)
```

```
## # A tibble: 6 × 6
##   name        species level       talent  audience judge
##   <fct>       <fct>   <fct>       <fct>      <dbl> <dbl>
## 1 gladly      bear    competitive singing        8     2
## 2 bunny       bunny   competitive magic          8     2
## 3 cuddly paws bunny   competitive dancing        8     1
## 4 flopsy      bunny   competitive singing       10     1
## 5 lfb         bunny   competitive comedy         8     1
## 6 shadow      bunny   competitive magic          9     1
```

Q2c

ANSWER: Put your answer here. [Word count: 105]

The difference in the outputs arises from the way missing values are treated in base R and tidyverse. In base R, during logical operations involving NA values on the conditions “judge” and “audience”, the result will also be NA for rows where the operands contain NA. These rows are not removed, and will appear in the output. On the contrary, tidyverse functions like filter() excludes rows by default where the filtering condition evaluates to NA, so if either the condition on “judge” or “audience” evaluates to NA for a row, that row is excluded from the filtered output, leading to fewer rows in the output.

Q2d

```
# Put your code here
remove_NA <- !is.na(d$judge) & !is.na(d$audience)
d[(d$judge==1 | d$judge==2) & d$audience>=8 & remove_NA,]
```

```
## # A tibble: 6 × 6
##   name        species level       talent  audience judge
##   <fct>       <fct>   <fct>       <fct>      <dbl> <dbl>
## 1 gladly      bear    competitive singing        8     2
## 2 bunny       bunny   competitive magic          8     2
## 3 cuddly paws bunny   competitive dancing        8     1
## 4 flopsy      bunny   competitive singing       10     1
## 5 lfb         bunny   competitive comedy         8     1
## 6 shadow      bunny   competitive magic          9     1
```

Q3

Q3a

```
# Put your code here
dshort <- d %>%
  select(-c(judge,audience))
dshort
```

```
## # A tibble: 42 × 4
##    name        species level       talent
##    <fct>       <fct>   <fct>       <fct>
##  1 gladly      bear    competitive singing
##  2 gladly      bear    fun         dancing
##  3 panda       bear    fun         dancing
##  4 snowy       bear    fun         singing
##  5 bunny       bunny   competitive magic
##  6 bunny       bunny   fun         comedy
##  7 cuddly paws bunny   competitive dancing
##  8 cuddly paws bunny   fun         instrument
##  9 flopsy      bunny   competitive singing
## 10 flopsy      bunny   fun         instrument
## # ℹ 32 more rows
```

Q3b

```
# Put your code here
d2 <- dshort %>%
  pivot_wider(names_from = "level",values_from = "talent")
d2
```

```
## # A tibble: 23 × 4
##    name        species competitive fun
##    <fct>       <fct>   <fct>       <fct>
##  1 gladly      bear    singing     dancing
##  2 panda       bear    <NA>        dancing
```

```
##  3 snowy       bear    <NA>        singing
##  4 bunny       bunny   magic       comedy
##  5 cuddly paws bunny   dancing     instrument
##  6 flopsy      bunny   singing     instrument
##  7 giganticky  bunny   magic       dancing
##  8 lfb         bunny   comedy      singing
##  9 pink bunny  bunny   instrument  singing
## 10 shadow      bunny   magic       other
## # ℹ 13 more rows
```

Q3c

```
# optional code here
d2_modify <- d %>%
  pivot_wider(names_from = "level",values_from = "talent")
d2_modify
```

```
## # A tibble: 42 × 6
##    name        species audience judge competitive fun
##    <fct>       <fct>      <dbl> <dbl> <fct>       <fct>
##  1 gladly      bear           8     2 singing     <NA>
##  2 gladly      bear           9    NA <NA>        dancing
##  3 panda       bear           8    NA <NA>        dancing
##  4 snowy       bear           7    NA <NA>        singing
##  5 bunny       bunny          8     2 magic       <NA>
##  6 bunny       bunny         10    NA <NA>        comedy
##  7 cuddly paws bunny          8     1 dancing     <NA>
##  8 cuddly paws bunny          9    NA <NA>        instrument
##  9 flopsy      bunny         10     1 singing     <NA>
## 10 flopsy      bunny         10    NA <NA>        instrument
## # ℹ 32 more rows
```

ANSWER: Put your answer here. [Word count: 106]

The difference in the outputs when using d and dshort with the pivot_wider function is indeed due to the presence of NA values. When using a dataset like d that includes NA values, pivot_wider can lead to more outputs with NA because it treats NA as a valid level of a factor, creating additional columns for NA values. On the other hand, dshort does include columns where there are NA values. However, pivot_wider will generate NA values in the output if there is a mismatch in key-value pairs. This results in fewer outputs and a smaller tibble because there are fewer factor levels to pivot on.

Q3d

```
# Put your code here
d2 %>%
  filter(competitive == fun | is.na(competitive) | is.na(fun))
```

```
## # A tibble: 6 × 4
##   name     species competitive fun
```

```
##   <fct>    <fct>   <fct>       <fct>
## 1 panda    bear    <NA>        dancing
## 2 snowy    bear    <NA>        singing
## 3 tweak    cat     singing     singing
## 4 barky    dog     comedy      <NA>
## 5 quackers duck    comedy      comedy
## 6 monkey   monkey  <NA>        singing
```

ANSWER: The names of the individuals who broke Rule 1 (i.e., that everybody needs to participate in both fun and competitive levels) are panda, snowy, barky and monkey. The names of the individuals who broke Rule 2 (i.e., that everybody must to do different kinds of talent at the fun and competitive levels) are tweak and quackers.

Q4

Q4a

```
# Put your code here
d <- d %>%
  arrange(name)
d
```

```
## # A tibble: 42 × 6
##    name        species  level       talent     audience judge
##    <fct>       <fct>    <fct>       <fct>         <dbl> <dbl>
##  1 barky       dog      competitive comedy            7     3
##  2 black       dog      competitive dancing           7     3
##  3 black       dog      fun         comedy            9    NA
##  4 bunny       bunny    competitive magic             8     2
##  5 bunny       bunny    fun         comedy           10    NA
##  6 cuddly paws bunny    competitive dancing           8     1
##  7 cuddly paws bunny    fun         instrument        9    NA
##  8 doggie      dog      competitive comedy           NA     2
##  9 doggie      dog      fun         singing           9    NA
## 10 douglas     hedgehog competitive singing           9     3
## # ℹ 32 more rows
```

Q4b

```
# Put your code here
d_full <- full_join(d, dd)
d_full
```

```
## # A tibble: 42 × 7
##    name        species  level       talent     audience judge durati
on
##    <fct>       <fct>    <fct>       <chr>         <dbl> <dbl>    <db
l>
##  1 barky       dog      competitive comedy            7     3     11.
6
##  2 black       dog      competitive dancing           7     3      5.
```

```
3
##  3 black       dog      fun         comedy            9    NA      9.
9
##  4 bunny       bunny    competitive magic             8     2      8.
9
##  5 bunny       bunny    fun         comedy           10    NA      8.
3
##  6 cuddly paws bunny    competitive dancing           8     1      4.
7
##  7 cuddly paws bunny    fun         instrument        9    NA      5.
5
##  8 doggie      dog      competitive comedy           NA     2      9.
2
##  9 doggie      dog      fun         singing           9    NA      4.
1
## 10 douglas     hedgehog competitive singing           9     3      4.
4
## # ℹ 32 more rows
```

Q4c

```
# This code has been given to you, you just need to run it
dc <- cbind(d,dd)
head(dc)

##          name species       level  talent audience judge    name
   level
## 1       barky     dog competitive  comedy        7     3   barky com
petitive
## 2       black     dog competitive dancing        7     3   black
     fun
## 3       black     dog         fun  comedy        9    NA   bunny
     fun
## 4       bunny   bunny competitive   magic        8     2  doggie com
petitive
## 5       bunny   bunny         fun  comedy       10    NA     lfb com
petitive
## 6 cuddly paws   bunny competitive dancing        8     1 paw paw com
petitive
##   talent duration
## 1 comedy     11.6
## 2 comedy      9.9
## 3 comedy      8.3
## 4 comedy      9.2
## 5 comedy      8.9
## 6 comedy      9.0
```

ANSWER: Put your answer here. [Word count: 87]

The output from cbind() results in more columns, totaling 10, whereas full_join() yields only 7 columns. Additionally, the tibble after using cbind() contains 3

repeated columns: “name”, “level”, and “talent”, whereas the columns after using full_join() are all unique. These differences arise because cbind() simply combines tibbles, preserving the original row order from the first tibble and then appending the second tibble. In contrast, full_join() performs a relational join operation, combining rows based on matching key columns and ensuring that values in common columns are aligned correctly.

Q5

Q5a

```
# Put your code here
df %>%
  mutate(durType = case_when(duration>10 ~ "long",
                             duration<5 ~ "short",
                             TRUE ~ "medium"))
```

```
## # A tibble: 42 × 8
##    name        species  level       talent     audience judge durati
on durType
##    <chr>       <chr>    <chr>       <chr>         <dbl> <dbl>    <db
l> <chr>
##  1 barky       dog      competitive comedy            7     3     11.
6 long
##  2 black       dog      competitive dancing           7     3      5.
3 medium
##  3 black       dog      fun         comedy            9    NA      9.
9 medium
##  4 bunny       bunny    competitive magic             8     2      8.
9 medium
##  5 bunny       bunny    fun         comedy           10    NA      8.
3 medium
##  6 cuddly paws bunny    competitive dancing           8     1      4.
7 short
##  7 cuddly paws bunny    fun         instrument        9    NA      5.
5 medium
##  8 doggie      dog      competitive comedy           NA     2      9.
2 medium
##  9 doggie      dog      fun         singing           9    NA      4.
1 short
## 10 douglas     hedgehog competitive singing           9     3      4.
4 short
## # ℹ  32 more rows
```

Q5b

```
# Put your code here
ds <- df %>%
  group_by(talent) %>%
  summarise(medAud = round(median(audience,na.rm = TRUE),2),
```

```
            mnAud = round(mean(audience,na.rm = TRUE),2),
            sdAud = round(sd(audience,na.rm = TRUE),2),
            n = n(),
            sderrAud = round(sdAud/sqrt(length(audience)),3)) %>%
  ungroup()
ds
```

```
## # A tibble: 6 × 6
##   talent     medAud mnAud sdAud     n sderrAud
##   <chr>       <dbl> <dbl> <dbl> <int>    <dbl>
## 1 comedy        9    8.29  1.8      8    0.636
## 2 dancing       8    8     1.29     7    0.488
## 3 instrument    7    7.5   1.76     6    0.719
## 4 magic         8.5  7.33  2.73     6    1.12
## 5 other         7.5  7.5   3.54     2    2.50
## 6 singing       9    8.5   1.31    13    0.363
```

Q5c

ANSWER: Put your answer here. [Word count: 90]

Based on the mean audience ratings, “magic” is the least popular, and based on the median audience ratings, “instrument” is the least popular.

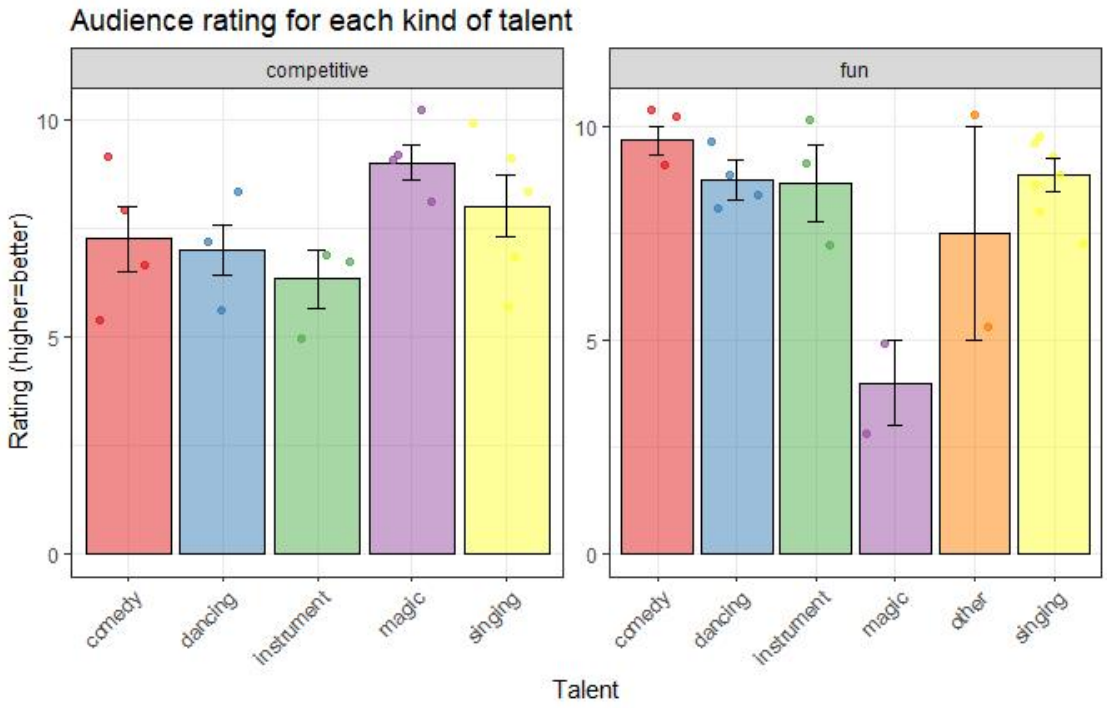
In the talent show data, despite the presence of other higher ratings, the mean rating for “magic” is heavily influenced by the extreme value 3, which pulls down the central tendency indicated by the mean. On the other hand, the median rating for “instrument” accurately captures central tendency by finding the middle value in the ordered dataset. By disregarding extreme values, the median provides a more robust measure of central tendency.

Q6

Q6a

```
# Put your code here
d6 %>%
  ggplot(mapping = aes(x = talent, y = mnAud, fill = talent)) +
  geom_col(alpha=0.5,show.legend=FALSE,colour="black") +
  geom_jitter(data=d_full,mapping=aes(x=talent,y=audience,color=talent),
              alpha=0.7 ,show.legend=FALSE) +
  geom_errorbar(aes(ymin = mnAud - sderrAud,
                    ymax = mnAud + sderrAud), width=0.2) +
  scale_fill_brewer(palette="Set1") +
  scale_colour_brewer(palette="Set1") +
  facet_wrap(~level, scales="free") +
  theme_bw() +
  labs(title = "Audience rating for each kind of talent",
       x = "Talent",
       y = "Rating (higher=better)") +
```

```
  theme(axis.text.x = element_text(angle = 45, hjust=1)) +
  scale_y_continuous(breaks = seq(0, 10, by = 5))
```



Q6b

ANSWER: Put your answer here. [Word count: 119]

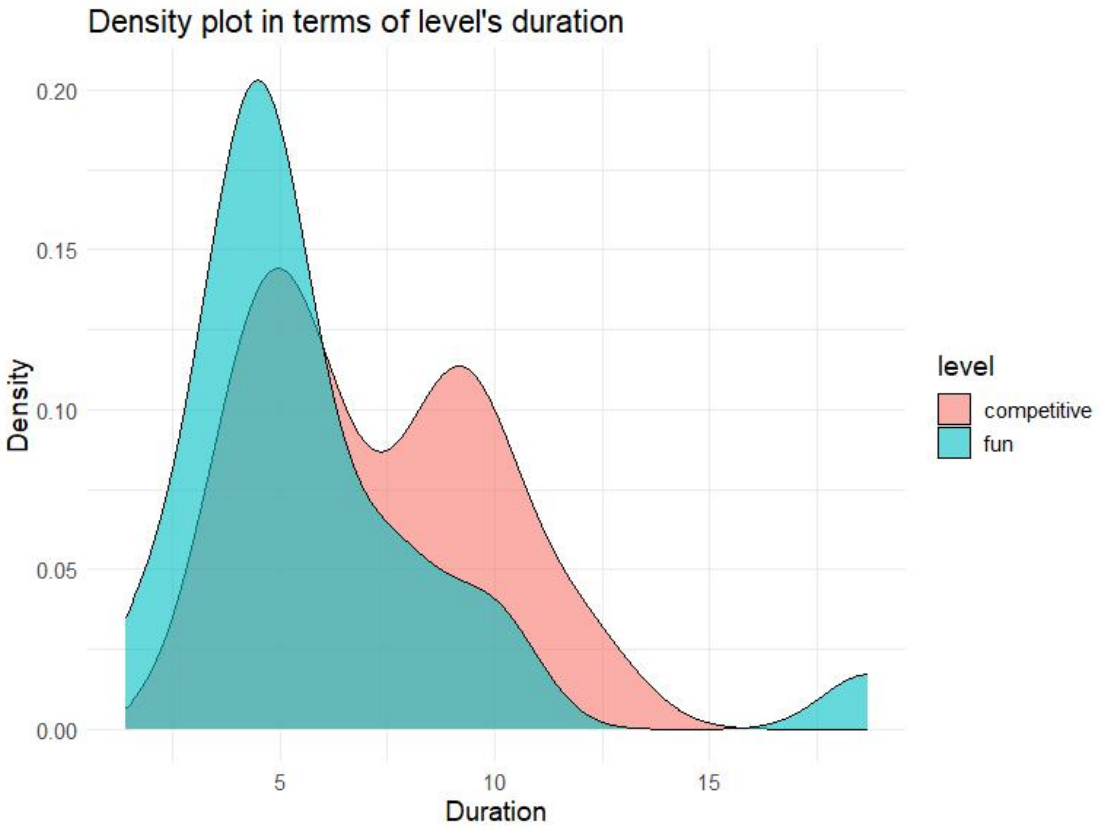
The audience ratings for different types of talent vary between the "competitive" and "fun" categories. In both categories, comedy seems to be the most appreciated talent, receiving the highest ratings. This could suggest that regardless of the context, comedy is universally enjoyed by the audience; Dancing and instrument also receive high ratings in both categories, indicating that these talents are well-received in both competitive and fun settings; Magic receives a lower rating in the fun category but fares better in the competitive category. This might suggest that the audience enjoys magic more when it's presented in a competitive, more intense and serious context; Singing receives moderate ratings in both categories, suggesting that it's neither exceptionally well-received nor poorly received.

Q7

Q7a

```
# Put your code here
df %>%
  ggplot(mapping = aes(x=duration,fill=level)) +
  geom_density(alpha=0.6,color = "black") +
  theme_minimal() +
  labs(title = "Density plot in terms of level's duration",
```

```
    x = "Duration",
    y = "Density") +
theme(text = element_text(size = 14))
```



Q7b

ANSWER: (1) Describe one new thing here. (2) Describe other new thing here. [Word count: 32]

(1) I changed the plot's overall theme to minimal style by adding theme_minimal() after geom_density.
(2) I adjusted the font size to 14 points of text elements by using theme(text = element_text(size = 14)).

Q7c

ANSWER: Put your answer here. [Word count: 129]

The density plot displays the distribution of duration for "fun" and "competitive" levels. The x-axis shows the duration, and the y-axis shows the density. The "fun" level shows a single peak around 5 minutes, indicating most "fun" levels have a duration close to this value. The "competitive" level shows a bimodal distribution with peaks around 5 and 10 minutes, suggesting two common durations for it. The higher peak for "competitive" levels is 5 minutes, similar to the peak for "fun" levels, implying a popular shorter duration for both level types. However, the second peak

for “competitive” levels around 10 minutes indicates some competitive levels have a longer duration. The plot also shows a small density for “fun” levels beyond 15 minutes, suggesting a few outliers with extremely long durations.

Q8

ANSWER: Put your answer here. [Word count: 71]

Gladly’s interpretation of the p-value is incorrect. A p-value is not a definitive proof of truth of the null hypothesis. Instead, it represents the probability of observing the test statistic if the null hypothesis is true. Moreover, changing the alpha threshold after obtaining the data, known as “p-hacking”, is problematic. It increases the Type I error rate and undermines the integrity of the statistical test, which can lead to incorrect conclusions.

Q9

```
# get the lowest score
lowest <- min(d$audience,na.rm=TRUE)
lowest

## [1] 3

# get the highest score
highest <- max(d$audience,na.rm=TRUE)
highest

## [1] 10
```

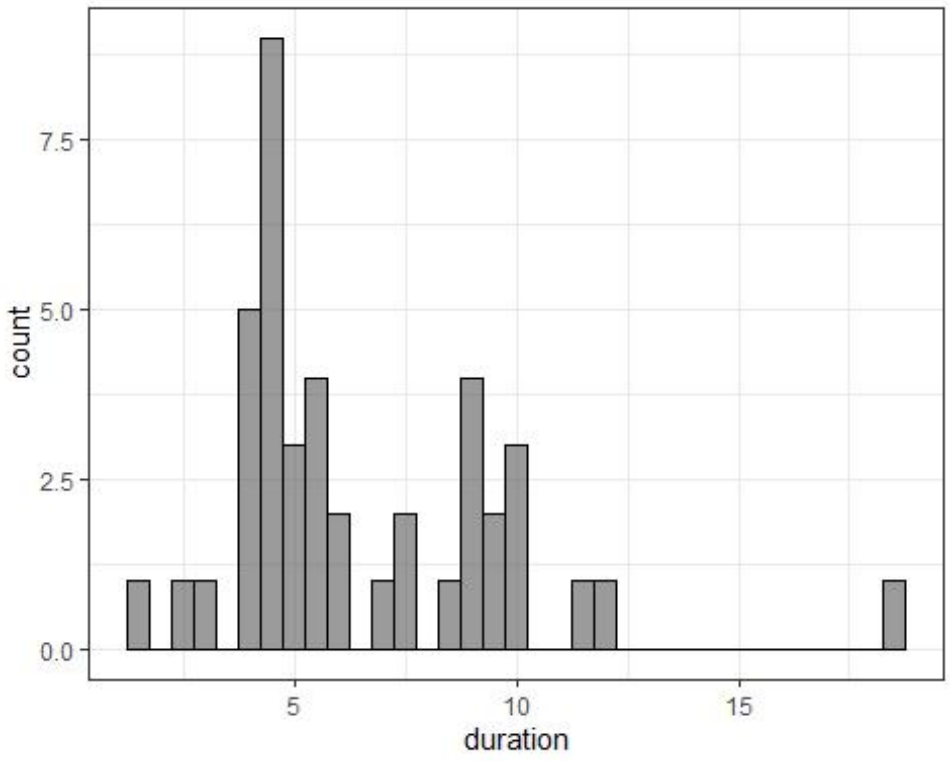
Q9a

```
# Put your code here
n <- 10
p <- 0.7

probLowest <- dbinom(x=lowest, size=n, prob=p)
probHighest <- dbinom(x=highest, size=n, prob=p)

probLowest <- round(probLowest * 100, 1)
probLowest

## [1] 0.9

probHighest <- round(probHighest * 100, 1)
probHighest

## [1] 2.8
```

ANSWER: The probability of the lowest score is 0.9% and the probability of the highest score is 2.8%.

Q9b

```r
# this code is given
df <- df %>%
  mutate(prob = pnorm(duration,mean=6.5,sd=3))

# you may add additional code here if it's useful to answer the questio
n
df %>%
  ggplot(mapping = aes(x=duration)) +
  geom_histogram(alpha=0.6,color = "black", binwidth = 0.5) +
  theme_bw()
```



ANSWER: Put your answer here. [Word count: 93]

The variable "prob" represents the probability of observing a "duration" value under the assumption that the true average duration is 6.5 minutes with a standard deviation of 3 minutes, while p-value is the probability of finding an observed test statistic, under the assumption that the null hypothesis is true. From the idea, we can get that p-value = 1 - prob, indicating that when prob is larger, p-value is smaller, vice versa. From the plot, we can identify that there is a data point significant different from previous averages, which is greater than 15.

Q9c

ANSWER: Put your answer here. [Word count: 123]

No, we cannot draw conclusions about the significance of the entire variable duration based on a single calculation combining only the individual prob values.

The prob values alone do not provide comprehensive insights into the distribution, variance, or other statistical characteristics of the duration data. They are just individual probabilities associated with each data point and do not reflect the overall behavior of the duration data. Moreover, this approach ignores the potential influence of other variables in the dataset on duration. These variables could also have significant interactions with duration that are not captured by looking at prob values alone. Therefore, extra information such as the descriptive statistics and the correlation coefficient could be helpful in analyzing the relationship between duration and prob.

Q10

Q10a

ANSWER: Put your answer here. [Word count: 90]

A sampling distribution represents the probability distribution of a statistic obtained from a large number of samples drawn from a population. The true distribution of audience ratings increases linearly from 0 to 10. This indicates that higher ratings are more probable than lower ones. Consequently, in a single timeslot consisting of 30 performances, we expect the range of ratings to be skewed towards higher values. Panel X best reflects this expected distribution with its increasing curve, aligning with the true distribution of ratings and the anticipated skew towards higher values.

Q10b

ANSWER: Put your answer here. [Word count: 97]

With a uniform distribution of audience ratings, ranging from 0 to 10 across 30 performances, each range is equally likely. The sampling distribution of the range is expected to be approximately symmetric and bell-shaped due to large sample sizes, making panel V the optimal choice.

If the true distribution changes, both the sampling distribution of the range and the mean will change accordingly. However, the mean is influenced by every value and is sensitive to outliers, while the range is determined by the extreme values and can vary widely if the true distribution has a large spread.

Q11

ANSWER: Put your answer here. Does not contribute to your word count limit.

In Bunnyland, everyone is hungry because they have all performed multiple talents, which consumed their energy, so they need tons of food.