Stats Homework 2

ANOVA

As an SE researcher you are evaluating different programming languages. For the next set of questions input the R code and interpret your findings.

a) The results of your first study compares Java, Python, and Ruby code based on the size of the programs in source (i.e. non-blank, non-commented) lines of code. Perform an ANOVA to determine whether there is an effect on size due to programming language. Use `lang-size.csv`.

```
# code goes here.
lang_size <- read.csv("C:\\Users\\delyar\\Desktop\\CS 567\\Stats HW 2\\lang-size.csv")
lang_size
```

```
##      lang sloc
## 1    java  207
## 2    java  296
## 3    java  348
## 4    java  309
## 5    java  231
## 6    java  228
## 7    java  318
## 8    java  212
## 9    java  284
## 10   java  267
## 11   java  354
## 12   java  262
## 13   java  259
## 14   java  342
## 15   java  252
## 16   java  299
## 17   java  312
## 18   java  285
## 19   java  266
## 20   java  333
## 21   java  280
## 22   java  283
## 23   java  368
## 24   java  407
## 25   java  325
## 26   java  339
## 27   java  255
## 28   java  327
## 29   java  268
## 30   java  315
```

```
## 31 python  401
## 32 python  391
## 33 python  331
## 34 python  370
## 35 python  415
## 36 python  419
## 37 python  340
## 38 python  382
## 39 python  319
## 40 python  387
## 41 python  455
## 42 python  438
## 43 python  388
## 44 python  449
## 45 python  437
## 46 python  404
## 47 python  352
## 48 python  390
## 49 python  446
## 50 python  424
## 51 python  370
## 52 python  291
## 53 python  366
## 54 python  294
## 55 python  337
## 56 python  381
## 57 python  366
## 58 python  356
## 59 python  395
## 60 python  387
## 61   ruby  188
## 62   ruby  227
## 63   ruby  208
## 64   ruby  267
## 65   ruby  303
## 66   ruby  311
## 67   ruby  287
## 68   ruby  226
## 69   ruby  278
## 70   ruby  188
## 71   ruby  269
## 72   ruby  178
## 73   ruby  198
## 74   ruby  239
## 75   ruby  176
## 76   ruby  309
## 77   ruby  176
## 78   ruby  228
## 79   ruby  197
## 80   ruby  326
## 81   ruby  280
## 82   ruby  239
## 83   ruby  286
## 84   ruby  286
```

```
## 85   ruby  272
## 86   ruby  258
## 87   ruby  283
## 88   ruby  361
## 89   ruby  191
## 90   ruby  246
```

```r
anova_results <- aov(lang_size$sloc ~ lang_size$lang)
summary(anova_results)
```

```
##                 Df Sum Sq Mean Sq F value Pr(>F)
## lang_size$lang  2 276056  138028    62.6 <2e-16 ***
## Residuals      87 191836    2205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
TukeyHSD(anova_results)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = lang_size$sloc ~ lang_size$lang)
##
## $`lang_size$lang`
##                   diff        lwr        upr     p adj
## python-java   88.33333   59.42296  117.24370 0.0000000
## ruby-java    -45.00000  -73.91037  -16.08963 0.0010476
## ruby-python -133.33333 -162.24370 -104.42296 0.0000000
```

```r
# in the results we see that the degree of freedom of the numerator is 2 and the degree of freedom of d
# the result of the test is 62.6
```

Report: Here we are having one independent variable which is the language and sloc which is the dependent variable. our Independent variable is categorical. Sloc is quantitative dependent variable. The independent variable has three levels. The *null hypothesis* (H_0) of ANOVA is that there is no difference among group means. The *alternative hypothesis* (H_a) is that at least one group differs from the overall mean of the dependent variable. The p-value is <2e-16.

1. The degrees of freedom (under column labelled `Df`) for the variable `lang_size$lang`. This is calculated as *(# of groups) - 1*, so in this case, there were 3 langs and the value is `3-1 = 2`.
2. The degrees of freedom for the residuals. This is calculated as *(# of total observations) - (# of groups)*. In this case, there were 90 observations and 3 groups, so this value is `90-3 = 87`.
3. The sum of squares (under column labelled `Sum Sq`) for the variable `lang_size$lang`. The sum of squares helps express the total variation that can be attributed to various factors; i.e. *sum of squares = treatment sum of squares (SST) + sum of squares of the residual error (SSE)*. In this case, the value is 276056.
4. The sum of squares of the residual. This value is 191836.
5. The mean square (under column labelled `Mean Sq`) for the variable `lang_size$lang`. This is calculated as *(sum of squares of treatment) / (Df of treatment)*, and allows you to determine whether there is a significant difference due to the treatment. The larger the ratio is, the more the treatments affect the outcome. In this case, it is calculated as `276056/2 = 138028`.

6. The mean square of the residuals. This is calculated as *(sum of squares of residuals) / (Df of residuals)*. In this case, it is calculated as `191836/87 = 2205`.
7. The overall F-statistic of the ANOVA model (under column labelled `F value`). This is calculated as *(mean square of treatment) / (mean square of residuals)*. In this case, it is calculated as `138028/2205 = 62.6`.
8. The p-value (under column labelled `Pr(>F)`) associated with the F-statistic with numerator `df = 2` and denominator `df = 87`. In this case, the p-value is <2e-16 , which is 2 X 10 ˆ -16 and well below either a `p<0.05`, `p<0.01`, or even `p<0.001` threshold for significance. The `***` stars beside this value also indicate where this fits on a significance range from 0 to 1.

Interpreting the results for our specific test, we see that the p-value in our ANOVA table is less than `p<0.05` and we therefore find sufficient evidence to **reject the null hypothesis** that all group means are equal. This means that we have sufficient evidence to say that the mean code size is not equal between the 3 programming languages.

Post-hoc ANOVA

If the p-value in the ANOVA output is less than 0.05, we reject the null hypothesis. This tells us that the mean value between each group is not equal. However, it doesn't tell us *which* groups differ from each other. In order to find this out, we must perform a post-hoc test. We can use the Tukey HSD

Here is how to interpret the Tukey results:

1. The adjusted p-value for the mean difference between group Java and Python is 0.0000000.
2. The adjusted p-value for the mean difference between group Java and Ruby is 0.0010476.
3. And so on for all combinations of pairs in the set.

The adjusted p-values that are less than 0.05. Therefore, we can conclude that there is a significant difference in mean.

b) In a subsequent study you measured the programming time (in hours) required to solve a program in Java, Python, and Ruby. This was a within subject study design: each participant solved the problem three times, and all participants solved the problem in the same order (Java, then Python, then Ruby). Perform an ANOVA to determine whether there is an effect due to programming language. Use `lang-time.csv`.

Two-way ANOVA

A two-way ANOVA (also called *multiple-factor ANOVA*) is used to determine whether or not there is a statistically significant difference between the means of three or more independent groups that have been split on two variables.

```
# code goes here
lang_time <- read.csv("C:\\Users\\delyar\\Desktop\\CS 567\\Stats HW 2\\lang-time.csv")
lang_time
```

```
##      lang participant times
## 1    java          P1  11.3
## 2    java          P2   7.8
## 3    java          P3  12.6
## 4    java          P4   6.3
```

```
## 5    java          P5   8.1
## 6    java          P6  10.1
## 7    java          P7   3.2
## 8    java          P8   7.3
## 9    java          P9   8.1
## 10   java         P10   7.9
## 11   java         P11   9.0
## 12   java         P12   8.7
## 13   java         P13   7.7
## 14   java         P14  11.6
## 15   java         P15   6.0
## 16   java         P16   8.1
## 17   java         P17  10.9
## 18   java         P18   9.1
## 19   java         P19   8.8
## 20   java         P20   9.8
## 21   java         P21   9.5
## 22   java         P22   8.7
## 23   java         P23  10.4
## 24   java         P24   9.7
## 25 python          P1   7.5
## 26 python          P2   5.1
## 27 python          P3   6.8
## 28 python          P4   5.7
## 29 python          P5   3.1
## 30 python          P6   6.8
## 31 python          P7   7.6
## 32 python          P8  11.2
## 33 python          P9  11.1
## 34 python         P10  10.2
## 35 python         P11   5.8
## 36 python         P12   4.3
## 37 python         P13   6.2
## 38 python         P14  10.5
## 39 python         P15   8.2
## 40 python         P16   8.8
## 41 python         P17   6.4
## 42 python         P18   4.9
## 43 python         P19   9.1
## 44 python         P20   7.3
## 45 python         P21   5.9
## 46 python         P22   8.0
## 47 python         P23   9.8
## 48 python         P24   9.8
## 49   ruby          P1   9.1
## 50   ruby          P2   6.2
## 51   ruby          P3   9.2
## 52   ruby          P4   6.9
## 53   ruby          P5   7.4
## 54   ruby          P6   9.7
## 55   ruby          P7   8.5
## 56   ruby          P8   7.1
## 57   ruby          P9   9.7
## 58   ruby         P10   6.2
```

```
## 59   ruby         P11   5.4
## 60   ruby         P12   7.7
## 61   ruby         P13   7.1
## 62   ruby         P14   6.9
## 63   ruby         P15   6.2
## 64   ruby         P16   5.3
## 65   ruby         P17   9.4
## 66   ruby         P18   3.7
## 67   ruby         P19   9.3
## 68   ruby         P20   4.9
## 69   ruby         P21   8.8
## 70   ruby         P22   4.4
## 71   ruby         P23   8.3
## 72   ruby         P24   5.7
```

```
aov <- aov(lang_time$times ~ lang_time$lang + lang_time$participant)
summary(aov)
```

```
##                       Df Sum Sq Mean Sq F value Pr(>F)
## lang_time$lang         2  33.32  16.661   4.583 0.0153 *
## lang_time$participant 23 110.62   4.809   1.323 0.2061
## Residuals             46 167.24   3.636
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
TukeyHSD(aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = lang_time$times ~ lang_time$lang + lang_time$participant)
##
## $`lang_time$lang`
##                   diff       lwr         upr     p adj
## python-java -1.2750000 -2.608038  0.05803792 0.0634715
## ruby-java   -1.5666667 -2.899705 -0.23362875 0.0177562
## ruby-python -0.2916667 -1.624705  1.04137125 0.8571070
##
## $`lang_time$participant`
##                diff       lwr      upr     p adj
## P10-P1  -1.20000000 -7.215583 4.815583 1.0000000
## P11-P1  -2.56666667 -8.582249 3.448916 0.9903303
## P12-P1  -2.40000000 -8.415583 3.615583 0.9957519
## P13-P1  -2.30000000 -8.315583 3.715583 0.9975609
## P14-P1   0.36666667 -5.648916 6.382249 1.0000000
## P15-P1  -2.50000000 -8.515583 3.515583 0.9929400
## P16-P1  -1.90000000 -7.915583 4.115583 0.9998494
## P17-P1  -0.40000000 -6.415583 5.615583 1.0000000
## P18-P1  -3.40000000 -9.415583 2.615583 0.8578064
## P19-P1  -0.23333333 -6.248916 5.782249 1.0000000
## P2-P1   -2.93333333 -8.948916 3.082249 0.9594576
## P20-P1  -1.96666667 -7.982249 4.048916 0.9997426
## P21-P1  -1.23333333 -7.248916 4.782249 0.9999999
```

```
## P22-P1  -2.26666667 -8.282249 3.748916 0.9979953
## P23-P1   0.20000000 -5.815583 6.215583 1.0000000
## P24-P1  -0.90000000 -6.915583 5.115583 1.0000000
## P3-P1    0.23333333 -5.782249 6.248916 1.0000000
## P4-P1   -3.00000000 -9.015583 3.015583 0.9497979
## P5-P1   -3.10000000 -9.115583 2.915583 0.9324188
## P6-P1   -0.43333333 -6.448916 5.582249 1.0000000
## P7-P1   -2.86666667 -8.882249 3.148916 0.9676893
## P8-P1   -0.76666667 -6.782249 5.248916 1.0000000
## P9-P1    0.33333333 -5.682249 6.348916 1.0000000
## P11-P10 -1.36666667 -7.382249 4.648916 0.9999995
## P12-P10 -1.20000000 -7.215583 4.815583 1.0000000
## P13-P10 -1.10000000 -7.115583 4.915583 1.0000000
## P14-P10  1.56666667 -4.448916 7.582249 0.9999941
## P15-P10 -1.30000000 -7.315583 4.715583 0.9999998
## P16-P10 -0.70000000 -6.715583 5.315583 1.0000000
## P17-P10  0.80000000 -5.215583 6.815583 1.0000000
## P18-P10 -2.20000000 -8.215583 3.815583 0.9986696
## P19-P10  0.96666667 -5.048916 6.982249 1.0000000
## P2-P10  -1.73333333 -7.748916 4.282249 0.9999661
## P20-P10 -0.76666667 -6.782249 5.248916 1.0000000
## P21-P10 -0.03333333 -6.048916 5.982249 1.0000000
## P22-P10 -1.06666667 -7.082249 4.948916 1.0000000
## P23-P10  1.40000000 -4.615583 7.415583 0.9999992
## P24-P10  0.30000000 -5.715583 6.315583 1.0000000
## P3-P10   1.43333333 -4.582249 7.448916 0.9999988
## P4-P10  -1.80000000 -7.815583 4.215583 0.9999367
## P5-P10  -1.90000000 -7.915583 4.115583 0.9998494
## P6-P10   0.76666667 -5.248916 6.782249 1.0000000
## P7-P10  -1.66666667 -7.682249 4.348916 0.9999825
## P8-P10   0.43333333 -5.582249 6.448916 1.0000000
## P9-P10   1.53333333 -4.482249 7.548916 0.9999959
## P12-P11  0.16666667 -5.848916 6.182249 1.0000000
## P13-P11  0.26666667 -5.748916 6.282249 1.0000000
## P14-P11  2.93333333 -3.082249 8.948916 0.9594576
## P15-P11  0.06666667 -5.948916 6.082249 1.0000000
## P16-P11  0.66666667 -5.348916 6.682249 1.0000000
## P17-P11  2.16666667 -3.848916 8.182249 0.9989262
## P18-P11 -0.83333333 -6.848916 5.182249 1.0000000
## P19-P11  2.33333333 -3.682249 8.348916 0.9970493
## P2-P11  -0.36666667 -6.382249 5.648916 1.0000000
## P20-P11  0.60000000 -5.415583 6.615583 1.0000000
## P21-P11  1.33333333 -4.682249 7.348916 0.9999997
## P22-P11  0.30000000 -5.715583 6.315583 1.0000000
## P23-P11  2.76666667 -3.248916 8.782249 0.9776127
## P24-P11  1.66666667 -4.348916 7.682249 0.9999825
## P3-P11   2.80000000 -3.215583 8.815583 0.9746074
## P4-P11  -0.43333333 -6.448916 5.582249 1.0000000
## P5-P11  -0.53333333 -6.548916 5.482249 1.0000000
## P6-P11   2.13333333 -3.882249 8.148916 0.9991388
## P7-P11  -0.30000000 -6.315583 5.715583 1.0000000
## P8-P11   1.80000000 -4.215583 7.815583 0.9999367
## P9-P11   2.90000000 -3.115583 8.915583 0.9637451
## P13-P12  0.10000000 -5.915583 6.115583 1.0000000
```

```
## P14-P12  2.76666667 -3.248916 8.782249 0.9776127
## P15-P12 -0.10000000 -6.115583 5.915583 1.0000000
## P16-P12  0.50000000 -5.515583 6.515583 1.0000000
## P17-P12  2.00000000 -4.015583 8.015583 0.9996674
## P18-P12 -1.00000000 -7.015583 5.015583 1.0000000
## P19-P12  2.16666667 -3.848916 8.182249 0.9989262
## P2-P12  -0.53333333 -6.548916 5.482249 1.0000000
## P20-P12  0.43333333 -5.582249 6.448916 1.0000000
## P21-P12  1.16666667 -4.848916 7.182249 1.0000000
## P22-P12  0.13333333 -5.882249 6.148916 1.0000000
## P23-P12  2.60000000 -3.415583 8.615583 0.9887603
## P24-P12  1.50000000 -4.515583 7.515583 0.9999973
## P3-P12   2.63333333 -3.382249 8.648916 0.9869924
## P4-P12  -0.60000000 -6.615583 5.415583 1.0000000
## P5-P12  -0.70000000 -6.715583 5.315583 1.0000000
## P6-P12   1.96666667 -4.048916 7.982249 0.9997426
## P7-P12  -0.46666667 -6.482249 5.548916 1.0000000
## P8-P12   1.63333333 -4.382249 7.648916 0.9999877
## P9-P12   2.73333333 -3.282249 8.748916 0.9803372
## P14-P13  2.66666667 -3.348916 8.682249 0.9850103
## P15-P13 -0.20000000 -6.215583 5.815583 1.0000000
## P16-P13  0.40000000 -5.615583 6.415583 1.0000000
## P17-P13  1.90000000 -4.115583 7.915583 0.9998494
## P18-P13 -1.10000000 -7.115583 4.915583 1.0000000
## P19-P13  2.06666667 -3.948916 8.082249 0.9994572
## P2-P13  -0.63333333 -6.648916 5.382249 1.0000000
## P20-P13  0.33333333 -5.682249 6.348916 1.0000000
## P21-P13  1.06666667 -4.948916 7.082249 1.0000000
## P22-P13  0.03333333 -5.982249 6.048916 1.0000000
## P23-P13  2.50000000 -3.515583 8.515583 0.9929400
## P24-P13  1.40000000 -4.615583 7.415583 0.9999992
## P3-P13   2.53333333 -3.482249 8.548916 0.9917184
## P4-P13  -0.70000000 -6.715583 5.315583 1.0000000
## P5-P13  -0.80000000 -6.815583 5.215583 1.0000000
## P6-P13   1.86666667 -4.148916 7.882249 0.9998862
## P7-P13  -0.56666667 -6.582249 5.448916 1.0000000
## P8-P13   1.53333333 -4.482249 7.548916 0.9999959
## P9-P13   2.63333333 -3.382249 8.648916 0.9869924
## P15-P14 -2.86666667 -8.882249 3.148916 0.9676893
## P16-P14 -2.26666667 -8.282249 3.748916 0.9979953
## P17-P14 -0.76666667 -6.782249 5.248916 1.0000000
## P18-P14 -3.76666667 -9.782249 2.248916 0.7238903
## P19-P14 -0.60000000 -6.615583 5.415583 1.0000000
## P2-P14  -3.30000000 -9.315583 2.715583 0.8864846
## P20-P14 -2.33333333 -8.348916 3.682249 0.9970493
## P21-P14 -1.60000000 -7.615583 4.415583 0.9999914
## P22-P14 -2.63333333 -8.648916 3.382249 0.9869924
## P23-P14 -0.16666667 -6.182249 5.848916 1.0000000
## P24-P14 -1.26666667 -7.282249 4.748916 0.9999999
## P3-P14  -0.13333333 -6.148916 5.882249 1.0000000
## P4-P14  -3.36666667 -9.382249 2.648916 0.8677824
## P5-P14  -3.46666667 -9.482249 2.548916 0.8366410
## P6-P14  -0.80000000 -6.815583 5.215583 1.0000000
## P7-P14  -3.23333333 -9.248916 2.782249 0.9034927
```

```
## P8-P14  -1.13333333 -7.148916 4.882249 1.0000000
## P9-P14  -0.03333333 -6.048916 5.982249 1.0000000
## P16-P15  0.60000000 -5.415583 6.615583 1.0000000
## P17-P15  2.10000000 -3.915583 8.115583 0.9993140
## P18-P15 -0.90000000 -6.915583 5.115583 1.0000000
## P19-P15  2.26666667 -3.748916 8.282249 0.9979953
## P2-P15  -0.43333333 -6.448916 5.582249 1.0000000
## P20-P15  0.53333333 -5.482249 6.548916 1.0000000
## P21-P15  1.26666667 -4.748916 7.282249 0.9999999
## P22-P15  0.23333333 -5.782249 6.248916 1.0000000
## P23-P15  2.70000000 -3.315583 8.715583 0.9827975
## P24-P15  1.60000000 -4.415583 7.615583 0.9999914
## P3-P15   2.73333333 -3.282249 8.748916 0.9803372
## P4-P15  -0.50000000 -6.515583 5.515583 1.0000000
## P5-P15  -0.60000000 -6.615583 5.415583 1.0000000
## P6-P15   2.06666667 -3.948916 8.082249 0.9994572
## P7-P15  -0.36666667 -6.382249 5.648916 1.0000000
## P8-P15   1.73333333 -4.282249 7.748916 0.9999661
## P9-P15   2.83333333 -3.182249 8.848916 0.9713049
## P17-P16  1.50000000 -4.515583 7.515583 0.9999973
## P18-P16 -1.50000000 -7.515583 4.515583 0.9999973
## P19-P16  1.66666667 -4.348916 7.682249 0.9999825
## P2-P16  -1.03333333 -7.048916 4.982249 1.0000000
## P20-P16 -0.06666667 -6.082249 5.948916 1.0000000
## P21-P16  0.66666667 -5.348916 6.682249 1.0000000
## P22-P16 -0.36666667 -6.382249 5.648916 1.0000000
## P23-P16  2.10000000 -3.915583 8.115583 0.9993140
## P24-P16  1.00000000 -5.015583 7.015583 1.0000000
## P3-P16   2.13333333 -3.882249 8.148916 0.9991388
## P4-P16  -1.10000000 -7.115583 4.915583 1.0000000
## P5-P16  -1.20000000 -7.215583 4.815583 1.0000000
## P6-P16   1.46666667 -4.548916 7.482249 0.9999982
## P7-P16  -0.96666667 -6.982249 5.048916 1.0000000
## P8-P16   1.13333333 -4.882249 7.148916 1.0000000
## P9-P16   2.23333333 -3.782249 8.248916 0.9983619
## P18-P17 -3.00000000 -9.015583 3.015583 0.9497979
## P19-P17  0.16666667 -5.848916 6.182249 1.0000000
## P2-P17  -2.53333333 -8.548916 3.482249 0.9917184
## P20-P17 -1.56666667 -7.582249 4.448916 0.9999941
## P21-P17 -0.83333333 -6.848916 5.182249 1.0000000
## P22-P17 -1.86666667 -7.882249 4.148916 0.9998862
## P23-P17  0.60000000 -5.415583 6.615583 1.0000000
## P24-P17 -0.50000000 -6.515583 5.515583 1.0000000
## P3-P17   0.63333333 -5.382249 6.648916 1.0000000
## P4-P17  -2.60000000 -8.615583 3.415583 0.9887603
## P5-P17  -2.70000000 -8.715583 3.315583 0.9827975
## P6-P17  -0.03333333 -6.048916 5.982249 1.0000000
## P7-P17  -2.46666667 -8.482249 3.548916 0.9940101
## P8-P17  -0.36666667 -6.382249 5.648916 1.0000000
## P9-P17   0.73333333 -5.282249 6.748916 1.0000000
## P19-P18  3.16666667 -2.848916 9.182249 0.9187978
## P2-P18   0.46666667 -5.548916 6.482249 1.0000000
## P20-P18  1.43333333 -4.582249 7.448916 0.9999988
## P21-P18  2.16666667 -3.848916 8.182249 0.9989262
```

```
## P22-P18  1.13333333 -4.882249 7.148916 1.0000000
## P23-P18  3.60000000 -2.415583 9.615583 0.7897612
## P24-P18  2.50000000 -3.515583 8.515583 0.9929400
## P3-P18   3.63333333 -2.382249 9.648916 0.7771750
## P4-P18   0.40000000 -5.615583 6.415583 1.0000000
## P5-P18   0.30000000 -5.715583 6.315583 1.0000000
## P6-P18   2.96666667 -3.048916 8.982249 0.9548129
## P7-P18   0.53333333 -5.482249 6.548916 1.0000000
## P8-P18   2.63333333 -3.382249 8.648916 0.9869924
## P9-P18   3.73333333 -2.282249 9.748916 0.7376130
## P2-P19  -2.70000000 -8.715583 3.315583 0.9827975
## P20-P19 -1.73333333 -7.748916 4.282249 0.9999661
## P21-P19 -1.00000000 -7.015583 5.015583 1.0000000
## P22-P19 -2.03333333 -8.048916 3.982249 0.9995736
## P23-P19  0.43333333 -5.582249 6.448916 1.0000000
## P24-P19 -0.66666667 -6.682249 5.348916 1.0000000
## P3-P19   0.46666667 -5.548916 6.482249 1.0000000
## P4-P19  -2.76666667 -8.782249 3.248916 0.9776127
## P5-P19  -2.86666667 -8.882249 3.148916 0.9676893
## P6-P19  -0.20000000 -6.215583 5.815583 1.0000000
## P7-P19  -2.63333333 -8.648916 3.382249 0.9869924
## P8-P19  -0.53333333 -6.548916 5.482249 1.0000000
## P9-P19   0.56666667 -5.448916 6.582249 1.0000000
## P20-P2   0.96666667 -5.048916 6.982249 1.0000000
## P21-P2   1.70000000 -4.315583 7.715583 0.9999756
## P22-P2   0.66666667 -5.348916 6.682249 1.0000000
## P23-P2   3.13333333 -2.882249 9.148916 0.9258166
## P24-P2   2.03333333 -3.982249 8.048916 0.9995736
## P3-P2    3.16666667 -2.848916 9.182249 0.9187978
## P4-P2   -0.06666667 -6.082249 5.948916 1.0000000
## P5-P2   -0.16666667 -6.182249 5.848916 1.0000000
## P6-P2    2.50000000 -3.515583 8.515583 0.9929400
## P7-P2    0.06666667 -5.948916 6.082249 1.0000000
## P8-P2    2.16666667 -3.848916 8.182249 0.9989262
## P9-P2    3.26666667 -2.748916 9.282249 0.8952016
## P21-P20  0.73333333 -5.282249 6.748916 1.0000000
## P22-P20 -0.30000000 -6.315583 5.715583 1.0000000
## P23-P20  2.16666667 -3.848916 8.182249 0.9989262
## P24-P20  1.06666667 -4.948916 7.082249 1.0000000
## P3-P20   2.20000000 -3.815583 8.215583 0.9986696
## P4-P20  -1.03333333 -7.048916 4.982249 1.0000000
## P5-P20  -1.13333333 -7.148916 4.882249 1.0000000
## P6-P20   1.53333333 -4.482249 7.548916 0.9999959
## P7-P20  -0.90000000 -6.915583 5.115583 1.0000000
## P8-P20   1.20000000 -4.815583 7.215583 1.0000000
## P9-P20   2.30000000 -3.715583 8.315583 0.9975609
## P22-P21 -1.03333333 -7.048916 4.982249 1.0000000
## P23-P21  1.43333333 -4.582249 7.448916 0.9999988
## P24-P21  0.33333333 -5.682249 6.348916 1.0000000
## P3-P21   1.46666667 -4.548916 7.482249 0.9999982
## P4-P21  -1.76666667 -7.782249 4.248916 0.9999535
## P5-P21  -1.86666667 -7.882249 4.148916 0.9998862
## P6-P21   0.80000000 -5.215583 6.815583 1.0000000
## P7-P21  -1.63333333 -7.648916 4.382249 0.9999877
```

```
## P8-P21    0.46666667 -5.548916 6.482249 1.0000000
## P9-P21    1.56666667 -4.448916 7.582249 0.9999941
## P23-P22  2.46666667 -3.548916 8.482249 0.9940101
## P24-P22  1.36666667 -4.648916 7.382249 0.9999995
## P3-P22    2.50000000 -3.515583 8.515583 0.9929400
## P4-P22   -0.73333333 -6.748916 5.282249 1.0000000
## P5-P22   -0.83333333 -6.848916 5.182249 1.0000000
## P6-P22    1.83333333 -4.182249 7.848916 0.9999148
## P7-P22   -0.60000000 -6.615583 5.415583 1.0000000
## P8-P22    1.50000000 -4.515583 7.515583 0.9999973
## P9-P22    2.60000000 -3.415583 8.615583 0.9887603
## P24-P23 -1.10000000 -7.115583 4.915583 1.0000000
## P3-P23    0.03333333 -5.982249 6.048916 1.0000000
## P4-P23   -3.20000000 -9.215583 2.815583 0.9113575
## P5-P23   -3.30000000 -9.315583 2.715583 0.8864846
## P6-P23   -0.63333333 -6.648916 5.382249 1.0000000
## P7-P23   -3.06666667 -9.082249 2.948916 0.9386109
## P8-P23   -0.96666667 -6.982249 5.048916 1.0000000
## P9-P23    0.13333333 -5.882249 6.148916 1.0000000
## P3-P24    1.13333333 -4.882249 7.148916 1.0000000
## P4-P24   -2.10000000 -8.115583 3.915583 0.9993140
## P5-P24   -2.20000000 -8.215583 3.815583 0.9986696
## P6-P24    0.46666667 -5.548916 6.482249 1.0000000
## P7-P24   -1.96666667 -7.982249 4.048916 0.9997426
## P8-P24    0.13333333 -5.882249 6.148916 1.0000000
## P9-P24    1.23333333 -4.782249 7.248916 0.9999999
## P4-P3    -3.23333333 -9.248916 2.782249 0.9034927
## P5-P3    -3.33333333 -9.348916 2.682249 0.8773436
## P6-P3    -0.66666667 -6.682249 5.348916 1.0000000
## P7-P3    -3.10000000 -9.115583 2.915583 0.9324188
## P8-P3    -1.00000000 -7.015583 5.015583 1.0000000
## P9-P3     0.10000000 -5.915583 6.115583 1.0000000
## P5-P4    -0.10000000 -6.115583 5.915583 1.0000000
## P6-P4     2.56666667 -3.448916 8.582249 0.9903303
## P7-P4     0.13333333 -5.882249 6.148916 1.0000000
## P8-P4     2.23333333 -3.782249 8.248916 0.9983619
## P9-P4     3.33333333 -2.682249 9.348916 0.8773436
## P6-P5     2.66666667 -3.348916 8.682249 0.9850103
## P7-P5     0.23333333 -5.782249 6.248916 1.0000000
## P8-P5     2.33333333 -3.682249 8.348916 0.9970493
## P9-P5     3.43333333 -2.582249 9.448916 0.8474230
## P7-P6    -2.43333333 -8.448916 3.582249 0.9949428
## P8-P6    -0.33333333 -6.348916 5.682249 1.0000000
## P9-P6     0.76666667 -5.248916 6.782249 1.0000000
## P8-P7     2.10000000 -3.915583 8.115583 0.9993140
## P9-P7     3.20000000 -2.815583 9.215583 0.9113575
## P9-P8     1.10000000 -4.915583 7.115583 1.0000000
```

Report: Because this is a two-way ANOVA, the ANOVA table provides results broken out by group (i.e. the independent variables). In this case, we can see that only the `lang_time$lang` factor has a statistically significant effect on the mean number of times. This result leads us to believe that changing the lang will impact significantly the mean time; and that changing `participant` would not have such an effect.

c) Your realized you should have counterbalanced, so you replicated the study from (b) which uses a crossover design to control for ordering. Each participant solved the problem in all three languages, but in each participant solved them in a different order. Perform an ANOVA to determine whether there is an effect due to programming language. Use `lang-time-crossover.csv`.

```
# code goes here
cross <- read.csv("C:\\Users\\delyar\\Desktop\\CS 567\\Stats HW 2\\lang-time-crossover.csv")
cross
```

```
##    participant treatment   lang times
## 1           P1        T1   java   6.4
## 2           P2        T1   java   8.3
## 3           P3        T1 python   7.0
## 4           P4        T1 python  10.5
## 5           P5        T1   ruby  10.6
## 6           P6        T1   ruby   4.0
## 7           P1        T2 python   8.2
## 8           P2        T2   ruby   5.5
## 9           P3        T2   java   7.7
## 10          P4        T2   ruby   7.5
## 11          P5        T2   java   7.0
## 12          P6        T2 python   4.4
## 13          P1        T3   ruby   5.7
## 14          P2        T3 python   7.9
## 15          P3        T3   ruby   8.8
## 16          P4        T3   java   9.5
## 17          P5        T3 python   8.0
## 18          P6        T3   java   8.0
```

```
aov <- aov(cross$times ~ cross$lang + cross$participant + cross$treatment)
summary(aov)
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## cross$lang         2  2.170   1.085   0.377  0.698
## cross$participant  5 26.100   5.220   1.812  0.217
## cross$treatment    2  5.623   2.812   0.976  0.418
## Residuals          8 23.047   2.881
```

Report:

All of the p-values are bigger than 0.05 so it means that the mean differences are not statistically significant.

d) You have some simulated results from an experiment that compared development time for Java, Python and Ruby, for subjects with low experience and high experience. Perform an ANOVA and identify which factors (language, experience) had a statistically significant effect. Also specify whether the interaction between programming language and experience was statistically significant or not. Use `lang-time-exp.csv`.

```
# code goes here

data <- read.csv("C:\\Users\\delyar\\Desktop\\CS 567\\Stats HW 2\\lang-time-exp.csv")
data
```

```
##      lang  exp times
## 1     java  low  11.0
## 2     java  low  10.6
## 3     java  low   8.3
## 4     java  low   9.8
## 5     java  low  11.6
## 6     java  low  11.8
## 7     java  low   8.6
## 8     java  low  10.3
## 9     java  low   7.7
## 10    java  low  10.5
## 11    java  low  13.2
## 12    java  low  12.5
## 13    java  low  10.5
## 14    java  low  13.0
## 15    java  low  12.5
## 16    java  low  11.2
## 17    java  low   9.1
## 18    java  low  10.6
## 19    java  low  12.9
## 20    java  low  12.0
## 21    java high   5.8
## 22    java high   2.6
## 23    java high   5.7
## 24    java high   2.8
## 25    java high   4.5
## 26    java high   6.3
## 27    java high   5.6
## 28    java high   5.3
## 29    java high   6.8
## 30    java high   6.5
## 31    java high   3.3
## 32    java high   6.8
## 33    java high   8.9
## 34    java high   7.4
## 35    java high   4.2
## 36    java high   4.1
## 37    java high   7.7
## 38    java high   3.5
## 39    java high   6.4
## 40    java high   5.7
## 41  python  low  12.2
## 42  python  low   8.5
## 43  python  low   8.3
## 44  python  low  11.7
## 45  python  low   8.1
## 46  python  low   9.9
## 47  python  low  10.5
## 48  python  low   9.4
## 49  python  low   8.6
## 50  python  low  11.3
## 51  python  low   9.2
## 52  python  low   9.3
## 53  python  low  12.7
```

```
## 54  python  low  14.3
## 55  python  low  11.0
## 56  python  low  11.6
## 57  python  low   8.2
## 58  python  low  11.1
## 59  python  low   8.7
## 60  python  low  10.6
## 61  python high   3.5
## 62  python high   5.1
## 63  python high   4.3
## 64  python high   6.7
## 65  python high   8.1
## 66  python high   8.4
## 67  python high   7.5
## 68  python high   5.0
## 69  python high   7.1
## 70  python high   3.5
## 71  python high   6.8
## 72  python high   3.1
## 73  python high   3.9
## 74  python high   5.6
## 75  python high   3.0
## 76  python high   8.3
## 77  python high   3.0
## 78  python high   5.1
## 79  python high   3.9
## 80  python high   9.0
## 81    ruby  low  10.2
## 82    ruby  low   8.6
## 83    ruby  low  10.4
## 84    ruby  low  10.4
## 85    ruby  low   9.9
## 86    ruby  low   9.3
## 87    ruby  low  10.3
## 88    ruby  low  13.4
## 89    ruby  low   6.6
## 90    ruby  low   8.9
## 91    ruby  low   8.2
## 92    ruby  low   8.6
## 93    ruby  low   9.1
## 94    ruby  low  11.3
## 95    ruby  low  10.2
## 96    ruby  low   6.2
## 97    ruby  low   5.8
## 98    ruby  low  10.8
## 99    ruby  low   9.3
## 100   ruby  low  11.5
## 101   ruby high   3.5
## 102   ruby high   5.8
## 103   ruby high   2.9
## 104   ruby high   6.4
## 105   ruby high   3.7
## 106   ruby high   6.1
## 107   ruby high   6.3
```

```
## 108   ruby high   1.7
## 109   ruby high   7.1
## 110   ruby high   7.3
## 111   ruby high   2.4
## 112   ruby high   7.3
## 113   ruby high   4.1
## 114   ruby high   4.4
## 115   ruby high   6.8
## 116   ruby high   5.2
## 117   ruby high   7.1
## 118   ruby high   6.5
## 119   ruby high   7.5
## 120   ruby high   6.9
```

```r
model <- aov(data$times ~ data$lang + data$exp)
summary(model)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## data$lang     2   11.1     5.6   1.716  0.184
## data$exp      1  663.2   663.2 204.369 <2e-16 ***
## Residuals   116  376.4     3.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
print("--------------------")
```

```
## [1] "--------------------"
```

```r
summary(aov(data$times ~ data$lang + data$exp + data$lang:data$exp))
```

```
##                     Df Sum Sq Mean Sq F value Pr(>F)
## data$lang            2   11.1     5.6   1.730  0.182
## data$exp             1  663.2   663.2 206.137 <2e-16 ***
## data$lang:data$exp   2    9.7     4.8   1.502  0.227
## Residuals          114  366.8     3.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Report:

A p-value less than 0.05 (typically <= 0.05) is statistically significant. The p-value for data$lang is 0.184 so it is bigger than 0.05 and$ $value for data$exp is <2e-16 so it is less than 0.05 and it is statistially significant. If we think that these two variables might interact to create a synergistic effect, we can switch to using a model that includes the possibility of this interaction effect as seen in top code. Although, as we can see in our ANOVA output table, the p-value for this interaction is not significant with a p-value of `0.227`. Therefore, it is unlikely that there is an interaction effect between these factors.

Part 3: Data analysis of an experiment

In this question, you'll analyze the raw data from an experiment and write up the results (similar to a publication).

The data is from a experiment to test whether statically typed languages (e.g. Java) or dynamically typed languages (e.g. Python) require more programming effort. The study evaluates the languages on two problems, a "small" problem and a "large" problem, to see if the results change based on the size of the problem. The study is a factorial design. The raw data from the experiment is available in this file: `lang-time-size.csv`.

Analyze the data and write up a short "results" section (as if it were a part of a paper) with your analysis of the data. This section should contain:

- Analysis of variance tables to determine if there are any interactions
- Interaction plot between the 2 factors
- Effect sizes for programming language for the "small" problem and for the "large" problem.
- I am not looking for a specific format, use your judgement about the best way to present this data to convey the results to a reader.

Results: In this study, we analyzed the effect of our two independent variables that are programming language and the problem size on the dependent variable which is the time of the program.In order to begin our analysis of the data, first we checked whether the times data is normal or not with the Shapiro test. From the output, the $p < 0.05$ result shows that we reject the null hypothesis, which means that the distribution of our data is significantly different from the normal distribution. This means that we are able to perform non-parametric tests on our data. In our analysis we have two variables that affect the times variable. Programming language and the problem size are both categorical variables. So we performed a Chi-Squared test method for determining if these two categorical variables have significant correlation between them. The result showed that p-value is 1, which indicates no strong correlation between the problem size and programming language factors. This makes sense because the two variables are independent from each other. Then we decided to draw the interaction plot between the two independent variables to see their interaction with each other and the dependent variable. By looking at the interaction plot, we figured out that for program sizes that are small, the program time in python language is much lower than Java language. However, for program sizes that are large, it is evident that the program time is smaller using Java as the programming language rather than Python. We also ran the Fisher's test and made sure that our two categorical variables are independent from each other. The odds ratio from Fisher's test is 1 so our null hypothesis cannot be rejected. This means that the effect sizes of the two factors are equal to each other. Then we ran the Anova test to find out whether the differences between groups of data are statistically significant. It works by analyzing the levels of variance within the groups through samples taken from each of them. According to the results of the Anova test, we can see that the programming language does not have statistically significant effect on times because its p-value is greater than 0.05. However we can see that the p-value of the problem size is less than 0.05 so it has a statistically significant relation with the times variable.From the Anova test results we also can see that the p-value for the interaction between our two independent variables (lang and size) is less than 0.05 meaning that there is an interaction effect between these factors.It is also notable that when we have more than two groups per variable it is better to use anova, if not, it is better using a t-test.

```r
# Code for analysis goes here.
my_data <- read.csv("C:\\Users\\delyar\\Desktop\\CS 567\\Stats HW 2\\lang-time-size.csv")
my_data
```

```
##     times  lang  size
## 1    14.0  java small
## 2    20.3  java small
## 3    11.6  java small
## 4    16.6  java small
## 5    15.0  java small
## 6     8.0  java small
## 7    13.1  java small
## 8    17.4  java small
```

```
## 9    12.5   java small
## 10    8.7   java small
## 11   16.4   java small
## 12   11.5   java small
## 13   13.7   java small
## 14    8.0   java small
## 15   18.8   java small
## 16   13.0   java small
## 17   12.9   java small
## 18   13.1   java small
## 19    8.3   java small
## 20   10.9   java small
## 21   18.5   java small
## 22   18.6   java small
## 23   11.4   java small
## 24   11.2   java small
## 25   17.0   java small
## 26   14.7   java small
## 27   15.9   java small
## 28   10.3   java small
## 29   14.3   java small
## 30   16.6   java small
## 31   15.3   java large
## 32   19.0   java large
## 33   30.6   java large
## 34   25.0   java large
## 35   26.7   java large
## 36   22.7   java large
## 37   17.1   java large
## 38   27.3   java large
## 39   13.9   java large
## 40    8.5   java large
## 41   21.9   java large
## 42   24.8   java large
## 43   38.9   java large
## 44   11.1   java large
## 45   16.3   java large
## 46   29.0   java large
## 47   10.0   java large
## 48   31.5   java large
## 49   24.7   java large
## 50   26.5   java large
## 51   24.5   java large
## 52   27.2   java large
## 53   17.1   java large
## 54   32.4   java large
## 55   22.3   java large
## 56   11.9   java large
## 57   13.4   java large
## 58   23.2   java large
## 59   28.3   java large
## 60   18.1   java large
## 61   10.2 python small
## 62    7.3 python small
```

```
## 63    6.6 python small
## 64    3.8 python small
## 65    3.0 python small
## 66    4.4 python small
## 67    7.7 python small
## 68   11.2 python small
## 69    6.0 python small
## 70   14.0 python small
## 71    2.9 python small
## 72    1.5 python small
## 73    3.7 python small
## 74    7.0 python small
## 75    9.2 python small
## 76    6.8 python small
## 77   10.4 python small
## 78   13.0 python small
## 79    5.1 python small
## 80   17.2 python small
## 81    9.0 python small
## 82   18.3 python small
## 83    9.6 python small
## 84    7.1 python small
## 85    7.1 python small
## 86    7.0 python small
## 87    7.7 python small
## 88   11.5 python small
## 89    9.5 python small
## 90    8.4 python small
## 91   20.8 python large
## 92   16.5 python large
## 93   43.8 python large
## 94   30.0 python large
## 95   25.2 python large
## 96   39.4 python large
## 97   25.6 python large
## 98   37.3 python large
## 99   19.0 python large
## 100   7.5 python large
## 101  26.7 python large
## 102  26.8 python large
## 103  17.3 python large
## 104  38.7 python large
## 105  35.6 python large
## 106  24.3 python large
## 107  28.6 python large
## 108  34.4 python large
## 109   6.3 python large
## 110  30.3 python large
## 111  20.0 python large
## 112  32.0 python large
## 113  28.1 python large
## 114  30.2 python large
## 115  29.7 python large
## 116  31.9 python large
```

```
## 117  14.9 python large
## 118  23.3 python large
## 119  35.4 python large
## 120  25.4 python large
```

```r
#Analyzing variance
anova_model <- aov(my_data$times ~ my_data$lang + my_data$size + my_data$lang:my_data$size)
summary(anova_model)
```

```
##                           Df Sum Sq Mean Sq F value   Pr(>F)    
## my_data$lang               1      3       3   0.085    0.772    
## my_data$size               1   5410    5410 133.246  < 2e-16 ***
## my_data$lang:my_data$size   1    811     811  19.968 1.84e-05 ***
## Residuals                116   4709      41                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
TukeyHSD(anova_model)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = my_data$times ~ my_data$lang + my_data$size + my_data$lang:my_data$size)
##
## $`my_data$lang`
##                   diff       lwr      upr     p adj
## python-java -0.3383333 -2.642417 1.965751 0.7716957
##
## $`my_data$size`
##                  diff       lwr       upr p adj
## small-large -13.42833 -15.73242 -11.12425     0
##
## $`my_data$lang:my_data$size`
##                                diff         lwr        upr     p adj
## python:large-java:large     4.860000   0.5715926   9.148407 0.0195978
## java:small-java:large      -8.230000 -12.5184074  -3.941593 0.0000120
## python:small-java:large   -13.766667 -18.0550741  -9.478259 0.0000000
## java:small-python:large   -13.090000 -17.3784074  -8.801593 0.0000000
## python:small-python:large -18.626667 -22.9150741 -14.338259 0.0000000
## python:small-java:small    -5.536667  -9.8250741  -1.248259 0.0056466
```

```r
print("---------------------------------------")
```

```
## [1] "---------------------------------------"
```

```r
#checking to see if data is normal
shapiro.test(my_data$times)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  my_data$times
## W = 0.95676, p-value = 0.0007014
```

```r
print("---------------------------------------")
```

```
## [1] "---------------------------------------"
```

```r
# Create a data frame from the main data set.
analysis.data <- data.frame(my_data$lang, my_data$size)

# Create a table with the needed variables.
analysis.data = table(my_data$lang, my_data$size)
print(analysis.data)
```

```
##
##           large small
##   java       30    30
##   python     30    30
```

```r
# Perform the Chi-Square test.
print(chisq.test(analysis.data))
```

```
##
##  Pearson's Chi-squared test
##
## data:  analysis.data
## X-squared = 0, df = 1, p-value = 1
```

```r
print("---------------------------------------")
```

```
## [1] "---------------------------------------"
```

```r
#Fisher's test
# create a dataframe
df <- data.frame("python" = c(30, 30), "java" = c(30, 30), row.names = c("small", "large"))
df
```

```
##       python java
## small     30   30
## large     30   30
```

```r
# run the test
fisher.test(df)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  df
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.4598626 2.1745627
## sample estimates:
## odds ratio
##          1
```

```
print("----------------------------------------")
```

```
## [1] "----------------------------------------"
```

```
interaction.plot(my_data$lang,
                 my_data$size,
                 my_data$times,
                 fun = mean,
                 ylab = "Program Times",
                 xlab = "Programming Language",
                 col = c("red", "blue"),
                 lty = 1,
                 lwd = 4, #line width
                 trace.label = "Program Size")
```