```
## R Code for data analysis of RM examination

# Step 1: Install and load packages
install.packages("dplyr")
install.packages("effsize")
install.packages("lsr")
install.packages("car")
install.packages("jtools")
install.packages("emmeans")
devtools::install_github("dustinfife/flexplot")


library(flexplot)
library(dplyr)
library(effsize)
library(lsr)
library(car)
library(readxl)
library(boot)
library(lme4)
library(tidyr)
library(ggplot2)
library(jtools)
library(emmeans)
library(sjPlot)

# Step 2: Import the dataset
data_rm <- read_excel("RM LISTA 2023 -2019.xlsx")
summary(data_rm)

# Step 3: Rename the variables
data_rm <- data_rm %>%
  rename(cmp = `CMP-N`,
         uni = UNIVERSIDAD,
         sede = SEDE,
         tipo = TIPO,
         esp = ESP,
         serum = `SERUMS-N`,
         Honors = `5TO_SUP`,
         enam = `ENAM-N`,
         Residency_Exam_score = `NOTA-F`,
         modalidad = MODALIDAD)

glimpse(data_rm) # Review the variables

# Step 4: Select the variables for the analysis
rm_var <- data_rm %>%
  select(ID, AÑO, cmp, uni, sede, tipo, esp, modalidad, serum, Honors, enam,
Residency_Exam_score)
glimpse(rm_var) # Now you got 30,371
table(rm_var$tipo)

# Step 5: Remove some variables
rm_var <- rm_var %>%
  filter(!uni %in% c("CONTINENTAL", "OTROS"))
glimpse(rm_var) # 20

rm_var <- rm_var %>%
  filter(!tipo %in% c("Sub."))
glimpse(rm_var)

rm_var <- rm_var %>%
  filter(Residency_Exam_score != 0)


# Step 6:Mix 2023
rm_var <- rm_var %>%
  mutate(AÑO = case_when(
```

```
    AÑO == "2023  CONR" | AÑO == "2023 UNIV" ~ "2023",  # Change specific observations
    TRUE ~ AÑO  # Keep all other observations as they are
  ))
table(rm_var$AÑO)
glimpse(rm_var)

# Step 7: Explore
rm_var <- rm_var %>%
  arrange(cmp)

glimpse(rm_var)

# Step 8: Remove data from CMPs that were bad asigned
rm_var <- rm_var %>%
  filter(!cmp > 99000)


# Step 9: Revise the database and transform enam and Honors
glimpse(rm_var)
table(rm_var$Honors)
table(rm_var$enam)

# For ENAM
rm_var <- rm_var %>%
  mutate(enam = case_when(
           enam == "1"   ~ "A",  # Change 1 to A
           enam == "1.5" ~ "B",  # Change 1.5 to B
           enam == "2"   ~ "C",  # Change 2 to C
           is.na(enam)   ~ "D",  # Change NA to D
           TRUE          ~ enam  # Mantain all
         ))
table(rm_var$enam) # Now transform it as factors
rm_var$enam <- factor(rm_var$enam, levels = c("C", "B", "A", "D"), ordered = TRUE)
table(rm_var$enam)
levels(rm_var$enam)

# for Honor
table(rm_var$Honors)
rm_var <- rm_var %>%
  mutate(Honors = replace_na(Honors, "NO")) %>%
  mutate(Honors = ifelse(Honors == "SI", "YES", Honors))
table(rm_var$Honors)
rm_var$Honors <- factor(rm_var$Honors, levels = c("NO", "YES"), ordered = TRUE)
levels(rm_var$Honors)

# Change sede of CONAREME
rm_var <- rm_var %>%
  mutate(sede = replace_na(sede, "CONAREME"))
glimpse(rm_var)

# Recode ENAM to make sense
rm_var <- rm_var %>%
  mutate(ENAM_score = case_when(
    enam == "D" ~ 1,
    enam == "A" ~ 2,
    enam == "B" ~ 3,
    enam == "C" ~ 4,
    TRUE ~ NA_real_  # In case there are other unexpected levels
  ))
rm_var$ENAM_score <- factor(rm_var$ENAM_score, levels = c("1", "2", "3", "4"), ordered =
TRUE)
table(rm_var$ENAM_score)
levels(rm_var$ENAM_score)
glimpse(rm_var)

# Create a new variables to compare CONAREME versus Non-CONAREME
table(rm_var$uni)
rm_var$cona <- ifelse(rm_var$uni == "CONAREME", "conareme", "non_conareme")
```

```
# Lets review the dataset and create tables
# Replace AÑO per variables of interest
glimpse(rm_var)
score_per_enam <- rm_var %>%
  group_by(AÑO) %>% # Por Año
  summarize(count = n(),
            perc = count/28872 * 100,
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.25),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.75),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score),) %>%
  arrange(AÑO)
print(score_per_enam, n=100)



glimpse(rm_var)
score_per_enam <- rm_var %>%
  group_by(ENAM_score) %>% # Por sede
  summarize(count = n(),
            perc = count/28872 * 100,
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.25),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.75),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score),) %>%
  arrange(desc(mean_rm))
print(score_per_enam, n=100)



score_per_enam <- rm_var %>%
  group_by(esp) %>% # Por modalidad
  summarize(count = n(),
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.10),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.90),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score)) %>%
  arrange(desc(mean_rm))
print(score_per_enam, n=100)
write.csv(score_per_enam, "score_per_enam.csv", row.names = FALSE)

score_per_enam <- rm_var %>%
  group_by(Honors) %>% # Por Honors
  summarize(count = n(),
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.25),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.75),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score),) %>%
  arrange(desc(mean_rm))
print(score_per_enam, n=100)

score_per_enam <- rm_var %>%
```

```
  group_by(ENAM_score) %>% # Por ENAM_score
  summarize(count = n(),
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.25),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.75),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score),) %>%
  arrange(desc(mean_rm))
print(score_per_enam, n=100)

score_per_enam <- rm_var %>%
  group_by(esp) %>% # Por cona
  summarize(count = n(),
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.25),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.75),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score),) %>%
  arrange(desc(median_rm))
print(score_per_enam, n=100)



# This will be different as we are only considering universities
score_per_enam <- rm_var %>%
  group_by(sede) %>% # Por uni
  summarize(count = n(),
            perc = n()/25432 *100,
            mean_rm = mean(Residency_Exam_score),
            sd_rm = sd(Residency_Exam_score),
            Q1_rm = quantile(Residency_Exam_score, 0.25),
            median_rm = median(Residency_Exam_score),
            Q3_rm = quantile(Residency_Exam_score, 0.75),
            max_rm = max(Residency_Exam_score),
            min_rm = min(Residency_Exam_score),) %>%
  arrange(desc(mean_rm))
print(score_per_enam, n=100)

# Phase 2: Lets create the graph
meddata <- read_excel("meddata.xlsx") # Load

glimpse(meddata) # Review

meddata <- meddata %>% # Change the names
  mutate(N = sprintf("%.2f%%", N/28872 * 100)) %>%
  mutate(`Medical specialty` = paste(`Medical specialty`, " (", N, ")", sep = ""))
meddata$`Medical specialty` <- factor(meddata$`Medical specialty`, levels =
meddata$`Medical specialty`[order(meddata$Md, decreasing = FALSE)])

# Create the Plot
ggplot_object <- ggplot(meddata, aes(y = `Medical specialty`, x = Md)) +
  geom_segment(aes(x = p10, xend = p90, y = `Medical specialty`, yend = `Medical
specialty`, colour = "Range (P10-P90)"),
               arrow = arrow(type = "closed", ends = "both", length = unit(0.05,
"inches"))) +
  geom_point(aes(x = Md, color = "Median"), size = 3) +
  geom_point(aes(x = Max, color = "Max score"), shape = 18, size = 3) +  # Using shape
18 for diamond
  geom_vline(aes(xintercept = 11.7, color = "Overall median"), linetype = "dashed", size
= 1) +
  geom_vline(aes(xintercept = 14.8, color = "Overall p90"), linetype = "dashed", size =
0.5) +
  geom_vline(aes(xintercept = 8.6, color = "Overall p10"), linetype = "dashed", size =
0.5) +
```

```
  labs(title = '', x = 'Residency Exam Score', y = 'Medical Specialty') +
  scale_colour_manual(values = c("Range (P10-P90)" = "black",
                                 "Median" = "orange",
                                 "Max score" = "blue",
                                 "Overall median" = "orange",
                                 "Overall p90" = "blue",
                                 "Overall p10" = "red"),
                      name = "Legend",
                      guide = guide_legend(override.aes = list(
                        linetype = c("solid", "blank", "solid", "blank", "dashed",
"dashed"),
                        shape = c(18, 10, 16, 3, 3, 3),
                        size = 3))) +
  theme_minimal() +
  scale_x_continuous(breaks = seq(from = floor(min(meddata$p10)), to =
ceiling(max(meddata$Max)), by = 1)) +
  theme(
    panel.grid.major = element_line(color = "gray", size = 0.5, linetype = "solid"),
    panel.grid.minor = element_line(color = "gray", size = 0.25, linetype = "solid"),
    panel.background = element_rect(fill = "white"),
    axis.text.x = element_text(angle = 90, hjust = 1),
    legend.position = "right"
  )

# Print the plot to the RStudio viewer
print(ggplot_object)



# Phase 3: Models
## Step 1 : Group everything by specialties Review models
rm_var <- rm_var %>%
  mutate(Grouped_Specialties = case_when(
    esp %in% c("CARDIOLOGIA", "CIRUGIA DE TORAX Y CARDIOVASCULAR", "CIRUGIA PLASTICA",
               "DERMATOLOGIA", "OFTALMOLOGIA", "CIRUGIA DE CABEZA, CUELLO Y
MAXILOFACIAL",
               "ENDOCRINOLOGIA", "GASTROENTEROLOGIA", "NEUROCIRUGIA", "UROLOGIA",
               "MEDICINA DEL DEPORTE", "OTORRINOLARINGOLOGIA") ~ "Top Specialties",
    esp %in% c("CIRUGIA ONCOLOGICA", "NEFROLOGIA", "NEUROLOGIA", "NEONATOLOGIA",
"NEUMOLOGIA",
               "HEMATOLOGIA", "ORTOPEDIA Y TRAUMATOLOGIA", "REUMATOLOGIA",
               "MEDICINA DE ENFERMEDADES INFECCIOSAS Y TROPICALES", "CIRUGIA
PEDIATRICA",
               "GENETICA MEDICA", "RADIOLOGIA", "CIRUGIA GENERAL", "MEDICINA
ONCOLOGICA",
               "PSIQUIATRIA", "MEDICINA FISICA Y DE REHABILITACION", "PEDIATRIA",
               "GINECOLOGIA Y OBSTETRICIA", "MEDICINA INTENSIVA") ~ "Medium
Specialties",
    TRUE ~ "Bottom specialties"
  ))

table(rm_var$Grouped_Specialties)

# Step 2: Do the ANOVA model
glimpse(rm_var)
model_out_aov <- aov(Residency_Exam_score ~ ENAM_score + Honors + AÑO + sede +
modalidad, data = rm_var)
summary(model_out_aov)
eta_sq_result <- etaSquared(model_out_aov)
print(eta_sq_result)


# Phase 3: Selecting the right mixed model

# AÑO - Group
# uni - Group
# sede - Group
# Grouped_Specialties - Group
```

```
# modalidad - Group
# cona  - Group
# Honor - individual
# ENAM_score - individual

# Models
glimpse(rm_var)

# Create the models to compare
mode_out1 <- lmer( # Fail to converge - Hence removed
  Residency_Exam_score ~ ENAM_score + Honors +
    (1 | Grouped_Specialties) +  #
    (1 | sede) + # Provincia, Capital o CONAREME
    (1 | modalidad) +
    (1 | uni) +
    (1 | AÑO),
  data = rm_var)

mode_out2 <- lmer( ## This one is the best model
  Residency_Exam_score ~ ENAM_score + Honors +
    (1 | Grouped_Specialties) +
    (1 | sede) +
    (1 | modalidad) +
    (1 | AÑO),
  data = rm_var)


mode_out3 <- lmer(
  Residency_Exam_score ~ ENAM_score + Honors +
    (1 | Grouped_Specialties) +
    (1 | cona) +
    (1 | AÑO),
  data = rm_var)

mode_out4 <- lmer(
  Residency_Exam_score ~ ENAM_score + Honors +
    (1 | esp) +
    (1 | cona) +
    (1 | AÑO),
  data = rm_var)

mode_out5 <- lmer( # Simpler model
  Residency_Exam_score ~ factor(ENAM_score) + Honors +
    (1 | Grouped_Specialties),
  data = rm_var)

# Compare models

anova(mode_out1,mode_out2, mode_out3, mode_out4,mode_out5) # best models 1 and 2
summary(mode_out2) # Review the model

ranef_1 <- ranef(mode_out2)
print(ranef_1) # Extract random effects


tab_model(mode_out2) # Print the model and review it characteristics

sessionInfo()
```