```
library(tidyverse)

#### 2.1
heavy_smoke = read_csv('~/Desktop/FALL 2018/Biostat I/Homework/data/HW3_export/
HeavySmoke.csv', col_types = cols()) %>%
  janitor::clean_names() %>%
  mutate(diff = bmi_6yrs - bmi_base)

n1 = nrow(heavy_smoke) # sample size
dbar = sum(heavy_smoke$diff) / n1 # mean change in BMI
dif_sq = sum((heavy_smoke$diff - dbar)^2) # sum of squares
s1 = sqrt(dif_sq / (n1 - 1)) # sample sd

t_heavy = dbar / (s1/sqrt(n1)) # test statistic
rej1 = qt(p = 0.975, df = n1 - 1)

# t.test(x = heavy_smoke$diff, alternative = 'two.sided')

heavy_smoke %>%
  rename(ID = id,
         'BMI at baseline' = bmi_base,
         'BMI at 6 years' = bmi_6yrs,
         'Change in BMI' = diff) %>%
  knitr::kable(caption = 'Change in BMI in women who quit smoking')

#### 2.2
never_smoke = read_csv('~/Desktop/FALL 2018/Biostat I/Homework/data/HW3_export/
NeverSmoke.csv', col_types = cols()) %>%
  janitor::clean_names() %>%
  mutate(diff = bmi_6yrs - bmi_base)

# first test for equality of variance
n2 = nrow(never_smoke) # sample size
dbar2 = sum(never_smoke$diff) / n2 # difference in BMI change between groups
dif_sq2 = sum((never_smoke$diff - dbar2)^2) # sum of squares
s2 = sqrt(dif_sq2 / (n2 - 1)) # sample sd

# test equality of variances
f = s1^2 / s2^2
f > qf(p = 0.975, df1 = n1 - 1, df2 = n2 - 1)

# we conclude equal variances between groups
s_pooled = sqrt(((n1 - 1)*s1^2 + (n2 - 1)*s2^2) / (n1 + n2 - 2))

t_twosample = (dbar - dbar2) / (s_pooled*sqrt((1/n1) + (1/n2))) # test statistics for
two sample test
rej2 = qt(p = 0.975, df = n1 + n2 - 2) # critical value for two sample test

# check hand calculations with built-in function
# t.test(heavy_smoke$diff, never_smoke$diff, alternative = 'two.sided', var.equal = T)

bind_cols(heavy_smoke, never_smoke) %>%
  select(diff, diff1) %>%
  rename('Heavy smokers' = diff,
         'Never smoked' = diff1) %>%
  knitr::kable(caption = '6-year change in BMI for female former smokers vs. never
smokers')

#### 2.3
CI = c(
  (dbar - dbar2) - rej2*(s_pooled*sqrt((1/n1) + (1/n2))),
  (dbar - dbar2) + rej2*(s_pooled*sqrt((1/n1) + (1/n2)))
)


#### 2.4.b
```

```
var1 = 2.0^2
var2 = 1.5^2
mu1 = 3.0
mu2 = 1.7

# sample size calculator
samplesize_calc = function(mu1, mu2, var1, var2, power = 0.8, sig.level = 0.05){
  return(2*((var1 + var2)*(qnorm(power) + qnorm(1 - (sig.level/2)))^2) / (mu1 -
mu2)^2)
}

# sample sizes
a0.05_p0.8 = ceiling(samplesize_calc(mu1, mu2, var1, var2))
a0.05_p0.9 = ceiling(samplesize_calc(mu1, mu2, var1, var2, power = 0.9))
a0.025_p0.8 = ceiling(samplesize_calc(mu1, mu2, var1, var2, sig.level = 0.025))
a0.025_p0.9 = ceiling(samplesize_calc(mu1, mu2, var1, var2, power = 0.9, sig.level =
0.025))

tibble(' ' = c('2.5% significance', '5% significance'),
       '80% power' = c(a0.025_p0.8, a0.05_p0.8),
       '90% power' = c(a0.025_p0.9, a0.05_p0.9)) %>%
  knitr::kable(caption = 'Total sample size requirements for the specified power and
significance levels')



#### 3.1
# import data
knee_data = read_csv('~/Desktop/FALL 2018/Biostat I/Homework/data/HW3_export/
Knee.csv', col_types = cols())

n = nrow(knee_data) # number of rows in knee data

# produce table of summary statistics
apply(X = knee_data, FUN = summary, MARGIN = 2) %>% # apply the summary function to
each column of the data
  do.call(bind_rows, .) %>% # take list produced by apply and turn into tibble
  mutate('Physical Health' = colnames(knee_data),
         `NA's` = ifelse(is.na(.$`NA's`), 0, `NA's`),
         'N' = ifelse(is.na(.$`NA's`), n, n - .$`NA's`)) %>%  # make a column
specifying what group each row corresponds to
  select(`Physical Health`, N, everything()) %>% # rearrange column order
  knitr::kable(caption = 'Summary statistics across') # make nice-looking table

# long format of knee data
knee_data = knee_data %>%
  gather(key = 'physical_status', value = 'days') %>%
  mutate(physical_status = fct_relevel(factor(physical_status), 'Below', 'Average',
'Above')) %>%
  filter(!is.na(days))

# boxplots
knee_data %>%
  ggplot(aes(x = physical_status, y = days)) +
  geom_boxplot() +
  labs(title = 'Number of days in physical therapy',
       x = 'Physical health status',
       y = 'Days')

#### 3.2
anova(lm(days~factor(physical_status), data = knee_data)) %>%
  knitr::kable()

f_CV = qf(0.99, 2, 22) # critical value
```

```
#### 3.3
# multiple testing adjustments
pairwise.t.test(knee_data$days, knee_data$physical_status, p.adj = 'bonferroni')
#Bonferroni
TukeyHSD(aov(days~factor(physical_status), data = knee_data)) # tukey
summary(multcomp::glht(aov(days~factor(physical_status), data = knee_data),
                       lincft = mcp(method = "Dunnett"))) # dunnett


#### 4.1
library(datasets)
data("UCBAdmissions")
UCBAdmissions = as.tibble(UCBAdmissions)

props = as.tibble(UCBAdmissions) %>%
  group_by(Gender, Admit) %>%
  summarise(Acceptances = sum(n)) %>%
  mutate('Proportion of total applications' = round(Acceptances / sum(Acceptances),
3)) %>%
  filter(Admit == 'Admitted') %>%
  select(-Admit)

props %>%
  knitr::kable()

# proportions and sample size for men and women applicants
prop.men = props$`Proportion of total applications`[2]
prop.women = props$`Proportion of total applications`[1]
n.women = 557 + 1278
n.men = 1198 + 1493

# 95% confidence intervals
men.ci = c(
  prop.men - qnorm(0.975)*sqrt((prop.men*(1 - prop.men))/n.men),
  prop.men + qnorm(0.975)*sqrt((prop.men*(1 - prop.men))/n.men)
)

women.ci = c(
  prop.women - qnorm(0.975)*sqrt((prop.women*(1 - prop.women))/n.women),
  prop.women + qnorm(0.975)*sqrt((prop.women*(1 - prop.women))/n.women)
)

# test for normality
(n.women)(prop.women)(1-prop.women) >=5 # women
(n.men)(prop.men)(1-prop.men) >=5 # men


#### 4.2

# p1, p2 = the proportion observed in groups 1 and 2
# n1, n2 = sample sizes in groups 1 and 2
# sig.level = significance (alpha) level. Default to 0.05
# cont.correction = logical (T/F) for whether to apply a continuity correction.
#                   Default is FALSE.
prop.z.test = function(p1, p2, n1, n2, sig.level = 0.05, cont.correction = FALSE){
  p.hat = (n1*p1 + n2*p2) / (n1 + n2) # pooled p
  p.bar = p1 - p2 # difference in proportions

  # conditional statements for whether to include a continuity correction
  if (cont.correction == F) {
    z = p.bar / sqrt(p.hat*(1 - p.hat)*(1/n1 + 1/n2))
  } else {
    z = (p.bar - (1/(2*n1) + 1/(2*n2))) / sqrt(p.hat*(1 - p.hat)*(1/n1 + 1/n2))
  }

  pval = 1 - pnorm(abs(z)) # calculate pvalue
```

```
  # return list of test statistics, pooled p, and pvalue
  return(list(
    z = z,
    phat = p.hat,
    p.val = pval
  ))
}

prop.z.test(prop.women, prop.men, n.women, n.men)
```