Methods 4

Will Tirone

Q1)

To get the tables, I used an online tool to extra the `.wikitable` object (https://wikitable2csv.ggor.de/).

a)

To select the sample, I just used the sample() function.

```
arenas = read.csv("arenas.csv")
arenas$Capacity = as.numeric(gsub(",", "", arenas$Capacity))

s = sample(arenas$Arena, size=40)

sample = arenas |>
  filter(Arena %in% s)

sample_cap = sample$Capacity
```

b)

```
# some values we need
n = 40
N = dim(arenas)[1]

# calculated values
srs_ybar = mean(sample_cap)
s2 = var(sample_cap)
var.hat = (1 - n/N) * s2 / n
```

```r
c.lower = srs_ybar - 1.96 * sqrt(var.hat)
c.upper = srs_ybar + 1.96 * sqrt(var.hat)

cat("A 95% C.I. for y-bar is : ", c.lower, c.upper )
```

```
A 95% C.I. for y-bar is :  5237.485 7689.051
```

We can see the average capacity $\bar{y} = 6463.2682927$. Since our sample is $>= 32$, we can use that as a rule of thumb that we can apply the CLT. So above, I've estimated that confidence interval using the finite population correction.

c)

Here, I'm reusing most of my code, but just subsetting the original arenas dataframe to include only where capacity $<= 9314$. Then, I take a sample of 40 from that, and apply the same estimates. The CLT would apply again since our sample size is still 40.

```r
small_arenas = arenas |>
  filter(Capacity <= 9314)

s = sample(small_arenas$Arena, size=40)

sample_small = small_arenas |> filter(Arena %in% s)

sample_small_cap = sample_small$Capacity

N = dim(small_arenas)[1]

# calculated values
srs_ybar = mean(sample_small_cap)
s2 = var(sample_small_cap)
var.hat = (1 - n/N) * s2 / n

c.lower = srs_ybar - 1.96 * sqrt(var.hat)
c.upper = srs_ybar + 1.96 * sqrt(var.hat)

cat("A 95% C.I. for y-bar is : ", c.lower, c.upper )
```

```
A 95% C.I. for y-bar is :  4576.492 5735.41
```

Q2)

a)

To take the stratified sample, I used $\frac{n_h}{n} \approx \frac{N_h}{N}$, then calculated each proportion following this. I did have to add 1 to the >= 2000 bracket to get the total to 40, but since in the slides it says approximately, I think that's okay (hopefully?).

Then, I subselected my data into the different strata, sample each of those strata, and combined them at the end which I fed into the svydesign() function.

```
set.seed(120)

n = 40

# adding arena time brackets
arenas = arenas |>
  mutate(time_bracket = case_when (
    Opened <= 1960 ~ "s1",
    Opened > 1960 & Opened <= 1979 ~ "s2",
    Opened >= 1980 & Opened <= 1999 ~ "s3",
    Opened >= 2000 ~ "s4"
  ))

# calculate n_h ratio
n1960 = round(51 / 382 * 40, 0) # 5
n1979 = round(123 / 382 * 40, 0) # 13
n1999 = round(118 / 382 * 40, 0) # 12
n2000 = round(90 / 382 * 40 + 1, 0) # 10

# filtering by strata
strata1960 = arenas |> filter(Opened <= 1960) # 51
strata1979 = arenas |> filter(Opened <= 1979 & Opened > 1960) # 123
strata1999 = arenas |> filter(Opened <= 1999 & Opened >= 1980) # 118
strata2000 = arenas |> filter(Opened >= 2000) # 90

N_strata1960 = 51
N_strata1979 = 123
N_strata1999 = 118
N_strata2000 = 90

# collect samples
```

```r
sample1960 = sample(strata1960$Arena, n1960)
sample1979 = sample(strata1979$Arena, n1979)
sample1999 = sample(strata1999$Arena, n1999)
sample2000 = sample(strata2000$Arena, n2000)

all_sample_names = c(sample1960, sample1979, sample1999, sample2000)
all_sample = arenas |> filter(Arena %in% all_sample_names)

# setting things for survey
wtvar = rep(0,n)
wtvar[1:5] = N_strata1960 / 5
wtvar[6:18] = N_strata1979 / 13
wtvar[19:30] = N_strata1999 / 12
wtvar[31:40] = N_strata2000 / 10

fpcvar = rep(0,n)
fpcvar[1:5] = N_strata1960
fpcvar[6:18] = N_strata1979
fpcvar[19:30] = N_strata1999
fpcvar[31:40] = N_strata2000

all_sample = all_sample |> bind_cols(wtvar= wtvar, fpcvar = fpcvar)

strat_survey = svydesign(~1, strata = ~time_bracket, weights = ~wtvar, fpc = ~fpcvar, data
```

b)

We can get the HT estimator using the following code. Again since the sample size >= 32 we can say that the CLT probably applies here.

```r
est = svymean(~Capacity, strat_survey)
c.lower = est - 1.96 * 634
c.upper = est + 1.96 * 634
cat("Estimate for average capacity : ", est, " And 95% CI: ", c.lower, c.upper)
```

```
Estimate for average capacity :  6978.484  And 95% CI:  5735.844 8221.124
```

c)

```r
set.seed(13000)

# adding arena time brackets
arenas = arenas |>
  mutate(time_bracket = case_when (
    Opened <= 1960 ~ "s1",
    Opened > 1960 & Opened <= 1979 ~ "s2",
    Opened >= 1980 & Opened <= 1999 ~ "s3",
    Opened >= 2000 ~ "s4"
  )) |>
  filter(Capacity <= 9314)

# calculate n_h ratio
n1960 = round(51 / 382 * 40, 0) # 5
n1979 = round(123 / 382 * 40, 0) # 13
n1999 = round(118 / 382 * 40, 0) # 12
n2000 = round(90 / 382 * 40 + 1, 0) # 10

# filtering by strata
strata1960 = arenas |> filter(Opened <= 1960) # 51
strata1979 = arenas |> filter(Opened <= 1979 & Opened > 1960) # 123
strata1999 = arenas |> filter(Opened <= 1999 & Opened >= 1980) # 118
strata2000 = arenas |> filter(Opened >= 2000) # 90

N_strata1960 = 51
N_strata1979 = 123
N_strata1999 = 118
N_strata2000 = 90

# collect samples
sample1960 = sample(strata1960$Arena, n1960)
sample1979 = sample(strata1979$Arena, n1979)
sample1999 = sample(strata1999$Arena, n1999)
sample2000 = sample(strata2000$Arena, n2000)

all_sample_names = c(sample1960, sample1979, sample1999, sample2000)
all_sample = arenas |> filter(Arena %in% all_sample_names)

# setting things for survey
```

```r
wtvar = rep(0,n)
wtvar[1:5] = N_strata1960 / 5
wtvar[6:18] = N_strata1979 / 13
wtvar[19:30] = N_strata1999 / 12
wtvar[31:40] = N_strata2000 / 10

fpcvar = rep(0,n)
fpcvar[1:5] = N_strata1960
fpcvar[6:18] = N_strata1979
fpcvar[19:30] = N_strata1999
fpcvar[31:40] = N_strata2000

all_sample = all_sample |> bind_cols(wtvar= wtvar, fpcvar = fpcvar)

strat_survey = svydesign(~1, strata = ~time_bracket, weights = ~wtvar, fpc = ~fpcvar, data
```

```r
est = svymean(~Capacity, strat_survey)
c.lower = est - 1.96 * 634
c.upper = est + 1.96 * 634
cat("Estimate for average capacity : ", est, " And 95% CI: ", c.lower, c.upper)
```

```
Estimate for average capacity :  4502.189  And 95% CI:  3259.549 5744.829
```

Q3)

We'll plug in values from our sample to the formula we derived in class for optimal allocation:

$$n_h = n \frac{N_h S_h / N}{\sum_h N_h S_h / N}$$

```r
n = 40
N = 382

# we have N_h, so just need S_h
S_1 = var(all_sample |>
            filter(time_bracket == 's1') |>
            select(Capacity) |>
            pull())
```

```
S_2 = var(all_sample |>
            filter(time_bracket == 's2') |>
            select(Capacity) |>
            pull())

S_3 = var(all_sample |>
            filter(time_bracket == 's3') |>
            select(Capacity) |>
            pull())

S_4 = var(all_sample |>
            filter(time_bracket == 's4') |>
            select(Capacity) |>
            pull())

denom = (1/N) * (S_1*N_strata1960 + S_2 * N_strata1979 + S_3 * N_strata1999 +
                   S_4 * N_strata2000)

# compute each n
n_1 = (n * (S_1 * N_strata1960)/N) / denom
n_2 = (n * (S_2 * N_strata1979)/N) / denom
n_3 = (n * (S_3 * N_strata1999)/N) / denom
n_4 = (n * (S_4 * N_strata2000)/N) / denom
```

Based on the above, and rounding so we get exactly 40, we have `n_1 = 2`, `n_2 = 26`, `n_3 = 9`, and `n_4` = 3, where each subscript corresponds to the strata it came from.

Q4)

a)

For both of these, we will use the optimized n_h we found in class.

The problem here is that we don't know N or N_h, but we do know the proportions. So, we will set $N_1 = 0.9$ for the houses that have internet, $N_2 = 0.1$, for the houses that don't have internet, and $N = 1$ for the total. We don't need specific numbers since just the ratio is important. Also, $c_h = 50$ for both strata and we assumed that the variances in the numerator and denominator were equal, so they cancel.

```
n_1 = 45000 * (.9 / sqrt(50)) / ((0.1)*sqrt(50) + (0.9)*sqrt(50))
n_2 = 45000 * (0.1 / sqrt(50)) / ((0.1)*sqrt(50) + (0.9)*sqrt(50))
```

So we see that the optimal $n_1 = 810$ and the optimal $n_2 = 90$

b)

Now the only difference is that $c_1 = \$10$ per unit and $c_2 = \$75$.

```r
n_1_online = 45000 * (.9 / sqrt(10)) / ((0.1)*sqrt(75) + (0.9)*sqrt(10))
n_2_inperson = 45000 * (.1 / sqrt(75)) / ((0.1)*sqrt(75) + (0.9)*sqrt(10))
```

And now we have many more internet samples at $n_1 = 3450$ and a few more households as well at $n_2 = 140$. Clearly this is the better option.

Q5)

a)

First, we note that we only have the sampling weights and not the FPC. However, we can group by each strata, count the units, and multiply by `TABTRUCKS` since the weights are just N_h/n_h. This gives us FPCs for each n_h, which I added to the dataframe.I also added a 1 to reach row as the response variable that we'll use to create the totals later.

```r
# find FPC
fpcs = trucks |>
  group_by(STRATUM) |>
  summarise(fpc = n() * TABTRUCKS) |>
  unique()

# attach the FPC for each stratum
trucks = trucks |>
  left_join(fpcs) |>
  mutate(truck_count = 1)

# make the survey
truck.survey = svydesign(id = ~1, strata = ~STRATUM, weights = ~TABTRUCKS,
                         fpc = ~fpc, data=trucks)
```

First, we estimate the total count. Of course, since we have no standard error, we can't make a confidence interval since we just have a spike at the value. I assume we have a very low standard error because we have a huge sample size.

```r
svytotal(~truck_count, truck.survey)
```

```
               total SE
truck_count 85174776  0
```

b)

Estimating the total miles driven, we get:

$$total = 1.11E + 12$$

and a 95% CI as :

$$1.11E + 12 \pm 1.96 * 6,491,502,012$$

And the above were produced from the following code:

```r
svytotal(~MILES_ANNL, truck.survey)
```

```
                total         SE
MILES_ANNL 1.1147e+12 6491502012
```

c)

And applying the same method to each truck type, we have one data frame showing the estimates, and the second showing the lower and upper bounds of the 95% CI's.

```r
svyby(~MILES_ANNL, ~TRUCKTYPE, truck.survey, svytotal)
```

```
  TRUCKTYPE   MILES_ANNL         se
1         1 428294502082 4708467481
2         2 541099850893 4407687988
3         3  41279084490  394930176
4         4  31752656137  346577586
5         5  72301789843  515880878
```

```r
svyby(~MILES_ANNL, ~TRUCKTYPE, truck.survey, svytotal) |>
  mutate(lower = MILES_ANNL - 1.96 * se,
         upper = MILES_ANNL + 1.96 * se) |>
  select(lower, upper)
```

```
         lower        upper
1 419065905818 437523098345
2 532460782436 549738919349
3  40505021345  42053147634
4  31073364069  32431948205
5  71290663321  73312916364
```

Q6)

a)

In class, we proved that:

$$\hat{t}_y^{HT} = \sum_{h=1}^{H} N_h \bar{y}_h$$

So I will use this fact (without proving it again). Of course, with only one sample in each of the strata other than N_5, in N_1 for example we just have

$$(1) \cdot \frac{y_1}{1}$$

Since there is only one element. However, for N_5 we just turn to the formula above by plugging in h=5. Then combining these, we get:

$$\hat{t}_y^{HT} = y_1 + y_2 + y_3 + y_4 + N_5 \bar{y}_5$$

b)

The first friend is correct - since each y_1 has a first-order inclusion probability of 1, there is no variance in the stratas other than 5. Thus,

$$\begin{aligned} Var(\hat{t}_y^{HT}) &= Var(y_1 + y_2 + y_3 + y_4 + N_5\bar{y}_5) \\ &= Var(N_5\bar{y}_5) \end{aligned}$$

Which is equal to the variance stated in the problem.

Q7)

$$\begin{aligned} E(\hat{t}_y^*) &= E[\sum_{i\in S, i\neq i^*} w_i y_i + w y_i^*] \\ &= \sum_{i=1}^{N} w_i y_i E(I_i) + E(w y_i^*) \\ &= t_y + w y_i^* \end{aligned}$$

Noting that above, the second term has no randomness since we've already said that we picked the sample. The first summation term we proceed with like usual in the HT estimator setting.

So its bias is:

$$Bias(\hat{t}_y^*) = t_y + w y_i^* - t_y = w y_i^*$$

Which seems to intuitively make sense - we're sort of tacking on this extra term, so of course we'll shift the expected value away from the true value of the total.